LSDS-1747
Version 4.0

Department of the Interior
U.S. Geological Survey

Landsat 8-9
Calibration and Validation (Cal/Val)
Algorithm Description Document (ADD)

Version 4.0

January 2021

USGS
science for a changing world

Landsat 8-9
Calibration and Validation (Cal/Val)
Algorithm Description Document (ADD)

January 2021

Document Owner:

CODY ANDERSON

Digitally signed by CODY ANDERSON
Date: 2021.01.29 15:14:32 -06'00'

Cody Anderson Date
Cal/Val Manager
U.S. Geological Survey

Approved By:

BRIAN SAUER

Digitally signed by BRIAN SAUER
Date: 2021.01.31 22:36:53 -06'00'

Brian Sauer Date
L9 Project Manager
U.S. Geological Survey

EROS
Sioux Falls, South Dakota

Executive Summary

This Landsat 8-9 Calibration and Validation (Cal/Val) Algorithm Description Document (ADD) details all atmospheric, radiometric, and geometric processing algorithms for the data processing and image assessment of the Landsat 8 (L8) and Landsat 9 (L9) sensors

Landsat represents the world's longest continuously acquired collection of space-based moderate-resolution land remote-sensing data. More than four decades of imagery provide a unique resource for those who work in agriculture, geology, forestry, regional planning, education, mapping, and global change research. Landsat images are also invaluable for emergency response and disaster relief.

This document is under the USGS L9 Project Configuration Control Board (CCB) control. Please submit changes to this document, as well as supportive material justifying the proposed changes, via Change Request (CR) to the Process and Change Management Tool.

Document History

Document Number	Document Version	Publication Date	Change Number
LSDS-1747	Version 1.0	January 2018	CCR 106
LSDS-1747	Version 2.0	September 2018	CCR 143
LSDS-1747	Version 3.0	October 2019	CCR 183
LSDS-1747	Version 4.0	January 2021	CR 20368

Contents

List of Figures

List of Tables

Section 1 Introduction

The Landsat mission is a joint mission formulated, implemented, and operated by the National Aeronautics and Space Administration (NASA) and the Department of the Interior (DOI) U.S. Geological Survey (USGS). Landsat is a remote-sensing satellite mission providing coverage of the Earth's land surfaces. The Landsat series of satellites continue the 40+ years of global data collection and distribution.

1.1 Background

The goal of Landsat is to continue the collection, archiving, and distribution of multispectral imagery affording global, synoptic, and repetitive coverage of the Earth's land surfaces at a scale where natural and human-induced changes can be detected, differentiated, characterized, and monitored over time. The Landsat programmatic goals are stated in the United States Code, Title 15 Chapter 82 "Land Remote Sensing Policy" (derived from the Land Remote Sensing Policy Act of 1992). This policy requires that the Landsat Project provide data into the future that are sufficiently consistent with previous Landsat data to allow the detection and quantitative characterization of changes in or on the surface of the Earth. The highly successful Landsat series of missions have provided satellite coverage of the Earth's continental surfaces since 1972. The data from these missions constitute the longest continuous record of Earth's surface as seen from space.

Landsat 8 (L8), launched on February 11, 2013, and Landsat 9 (L9), to be launched in December 2020, are the latest satellites in the 40-year history of the Landsat Program. L9 was designed as a rebuild of L8 as much as possible. While there are a few differences between them, the sensors on-board each mission can in most ways be considered the same. Areas of difference that impact Calibration Validation (Cal/Val) algorithms are addressed in each individual algorithm description. If an algorithm does not specify for which mission it is applicable, it is applicable for both missions. Before L8 or L9 data are made available, they are radiometrically and geometrically corrected using processing inputs from the Calibration Parameter File (CPF), Bias Parameter File (BPF), and Response Linearization Look-Up Table (RLUT). The Calibration Validation Team (CVT) ensures that these files are monitored and updated over the life of the missions.

The Image Assessment System (IAS) assesses data on-orbit and monitors changes temporally. The radiometric, geometric, and spatial performance of the Operational Land Imagers (OLI) and Thermal Infrared Sensors (TIRS) are continually monitored, characterized, and calibrated on-orbit. Data processed by the Landsat Product Generation System (LPGS) are also trended to a database for later analysis by the CVT.

The CVT monitors the performances of both L8 and L9 data daily by trending the results of radiometric and geometric algorithms processed on all data. Through regular evaluation of the stored results in the database, changes in instrument behavior can be monitored and corrected over time. The CVT monitors the changes in the sensors and

determines whether to update the CPF, BPF, and RLUT to create better image products, while maintaining a level of consistency for comparability through time.

1.2 Purpose and Scope

The primary purpose of this document is to explain the methods for the atmospheric, geometric, and radiometric characterization and calibration of the L8/L9 OLI and TIRS instruments implemented at the United States Geological Survey (USGS) Earth Resources Observation and Science (EROS) Center.

The scope of this document is a brief overview of the instruments and their data, followed by discussions of the design philosophy, data flow diagrams, and algorithm descriptions developed by the CVT for atmospheric, geometric, and radiometric characterization and calibration.

1.3 Document Organization

This document contains the following sections:

- Section 1 provides an introduction
- Section 2 provides an overview of the OLI and TIRS Instruments
- Section 3 provides a look at the image production processing flows
- Section 4 provides details for each algorithm
- Appendix A provides a list of acronyms
- The References section provides a list of reference documents

1.4 Terminology

In this document, characterization is the process of measuring and evaluating the geometric and radiometric performance of the OLI and TIRS. Calibration is the process of using the information obtained during characterization to update the calibration parameters associated with both instruments.

Section 2 Instrument Overviews

L8 / L9 are joint missions formulated by the National Aeronautics and Space Administration (NASA) and the USGS and continue the 40-plus years of global Earth land surface data collection and distribution provided by the Landsat series. L8 was originally named the Landsat Data Continuity Mission (LDCM) before commissioning and was a large departure in many ways (e.g., pushbroom sensors instead of whiskbroom, spectral band definitions, etc.) from previous Landsats. L9 was designed to be a rebuild of L8 as much as possible. Some differences between the two missions include baffles added to the L9 TIRS telescope to reduce straylight contamination seen in the L8 TIRS, a different Scene Select Mirror (SSM) encoder for L9 TIRS because of the failure of the L8 TIRS SSM, as well as selecting or building new components for L9 to replace unavailable components used for L8.

The Space Segment consists of Observatories launched into a 705-kilometer (km), 10:00 a.m. equatorial crossing sun-synchronous orbit with a 16-day repeat period following the Worldwide Reference System 2 (WRS-2). Prior to the launch of L9, Landsat 7 (L7) and L8 occupied the same orbit but were phased eight days apart, allowing an opportunity for acquiring a WRS-2 scene every eight days. After launch, L9 will take L7's previous orbital position. The L8 / L9 spacecrafts, developed by Northrup Grumman, each accommodate an OLI, developed by Ball Aerospace & Technologies Corporation (BATC), and TIRS, developed by NASA Goddard Space Flight Center (GSFC).

2.1 OLI

The OLI images the Earth in nine spectral bands that cover the Visible and Near-Infrared (VNIR) and Short Wavelength Infrared (SWIR) portions of the electromagnetic spectrum (Table 2-1). With L8 OLI, all bands are downlinked at 12-bit radiometric resolution while the OLI sensor itself acquires data at 14-bit resolution. The L9 OLI will downlink all 14 bits from the sensor. Pixels for eight bands will be 30 meters while pixels for the panchromatic band will be 15 meters.

The OLI is a pushbroom sensor supplied by Ball Aerospace. The telescope contains four mirrors with a front aperture stop that is 135 millimeters (mm). The Focal Plane Array (FPA) includes 14 Sensor Chip Assemblies (SCAs) that overlap for about 25 30-meter pixels, as shown in Figure 2-1, that are passively cooled. Each SCA contains 494 imaging detectors, with an additional 12 Video Reference Pixels (VRPs) that do not respond to light.

2.1.1 On-Board Calibrators

The OLI provides both internal calibration sources, such as lamps, to ensure radiometric accuracy, as well as capabilities to perform solar and lunar calibrations within the Field of View (FOV) constraints.

2.1.1.1 Solar Calibration and Linearity

The spacecraft must point the sun-viewing boresight at the sun and track it. To assure the calibration returns valid results, there must be a glint-free FOV for the diffuser, as defined in the Interface Control Document (ICD). During solar looks, the solar array is angled to prevent it from infringing on the glint-free FOV.

2.1.1.2 Lunar Calibration

The spacecraft must perform sweeps across the moon to image the moon on all 14 Focal Plane Modules (FPMs). Because the moon is only large enough to subtend on 1 FPM, it requires 14 sweeps across the moon (over multiple orbits, if necessary) with the spacecraft yawing to place the moon on each of the FPMs.

2.1.2 Geolocation Calibration Activities

The following sections describe the geolocation calibration activities that require spacecraft operations.

2.1.2.1 Star Field Calibration

A section of the sky that is visible to both star trackers and the instrument must be imaged to locate stars against the star catalog. This occurs during commissioning only.

2.1.2.2 Calibration Data

To perform calibration, the raw star tracker and gyro data for ground processing must be available.

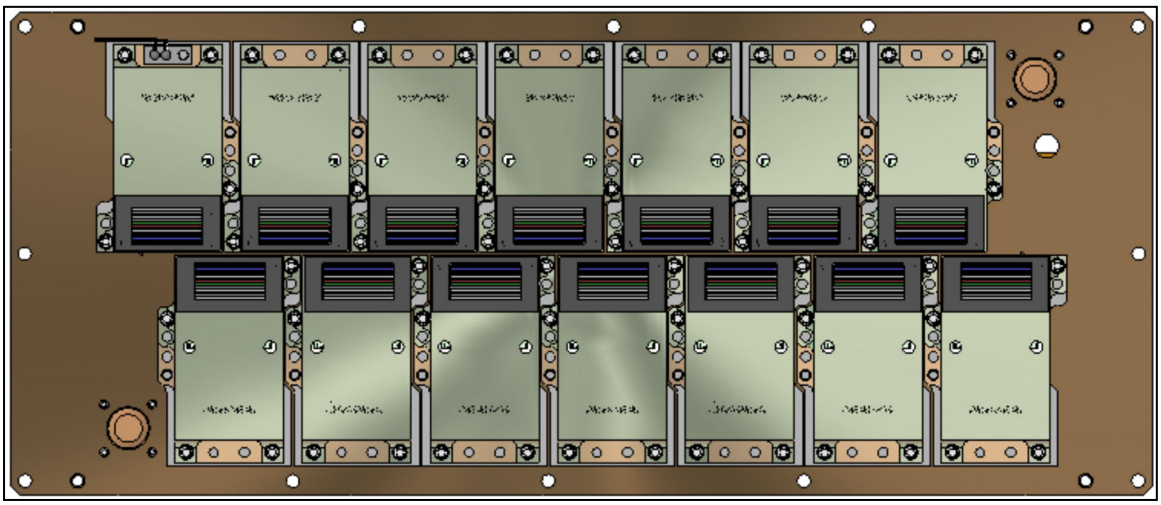

Figure 2-1. OLI Focal Plane Assembly

Band #	Band Center	Wavelength (nm)	Minimum Lower Band Edge (nm)	Maximum Upper Band Edge (nm)
1	Coastal Aerosol	443	435	451

Band #	Band Center	Wavelength (nm)	Minimum Lower Band Edge (nm)	Maximum Upper Band Edge (nm)
2	Blue	482	452	512
3	Green	561	533	590
4	Red	655	636	673
5	NIR	865	851	879
6	SWIR1	1609	1566	1651
7	SWIR2	2201	2107	2294
8	Panchromatic	590	503	676
9	Cirrus	1373	1363	1384

Table 2-1. Spectral Ranges of OLI Bands

2.2 TIRS

The TIRS is a two-band thermal imager with bands centered at 10.8 and 12 microns (Table 2-2). Both bands have a spatial resolution of 100 meters operating in a pushbroom method to achieve a 188-km swath width. The FPA, which includes three SCAs with Quantum Well Infrared Photometers (QWIPs), was built in-house at the NASA GSFC (shown in Figure 2-2. TIRS Focal Plane Assembly). TIRS collects and downlinks data in 12-bits. The FPA is cryo-cooled to 43K, with an optical assembly passively cooled to 180K. An SSM in the optical path allows calibration with two sources: a variable temperature blackbody and space views. The SSM encoder failed on L8, so the encoder vendor was changed for L9.

The optical design is a four-element refractive design with a 107.8 mm clear aperture. Three of the elements are based on germanium, and the fourth on zinc selenide. L8 TIRS had a significant straylight issue. To mitigate this issue, two baffles were added to the L9 optical design. TIRS has two spectral bands achieved through interference filters. The filters are thermally connected to the focal plane and operate at a somewhat higher temperature. Transmission characteristics are tailored for each band. Very good out-of-band rejection is required to perform precise spectral radiometry, and the in-band transmission must be high enough to meet the detector sensitivity goals. In addition, filter placement must accommodate a 2.5-second simultaneity requirement between 10.8 and 12 um measurements, and all data must be collected within 170 rows of detector pixels.

TIRS relies on QWIP detectors coupled with existing Indigo 9803 640 x 512-pixel Read-Out Integrated Circuits (ROICs) to give the previously mentioned 185 km swath in 3 arrays with 35-pixel overlap between arrays.

2.2.1 Onboard Calibrator

A key component for the TIRS is the onboard calibrator. The calibrator is a curved-plate blackbody with V-grooves to improve emissivity. The design and coating is very similar to that used for the Moderate Resolution Imaging Spectroradiometer (MODIS) to give high emissivity and controllable temperature. The output from the blackbody is National Institute for Standards and Technology (NIST) traceable and capable of providing

sources of two temperatures between 265 and 330 K within 2 orbits. Set point control of the blackbody is 2 K, with the capability to change the temperature by 6 K per half orbit.

2.2.2 Scene-Select Mirror

An SSM rotates around the optical axis on a 45-degree plane to provide the telescope with a view to nadir (Earth), space (cold calibration "target"), and on-board blackbody (hot calibration target). The mirror is based on a solid aluminum blank diamond turned flat and super polished. The size of the mirror is 206.5 x 148.5 mm.

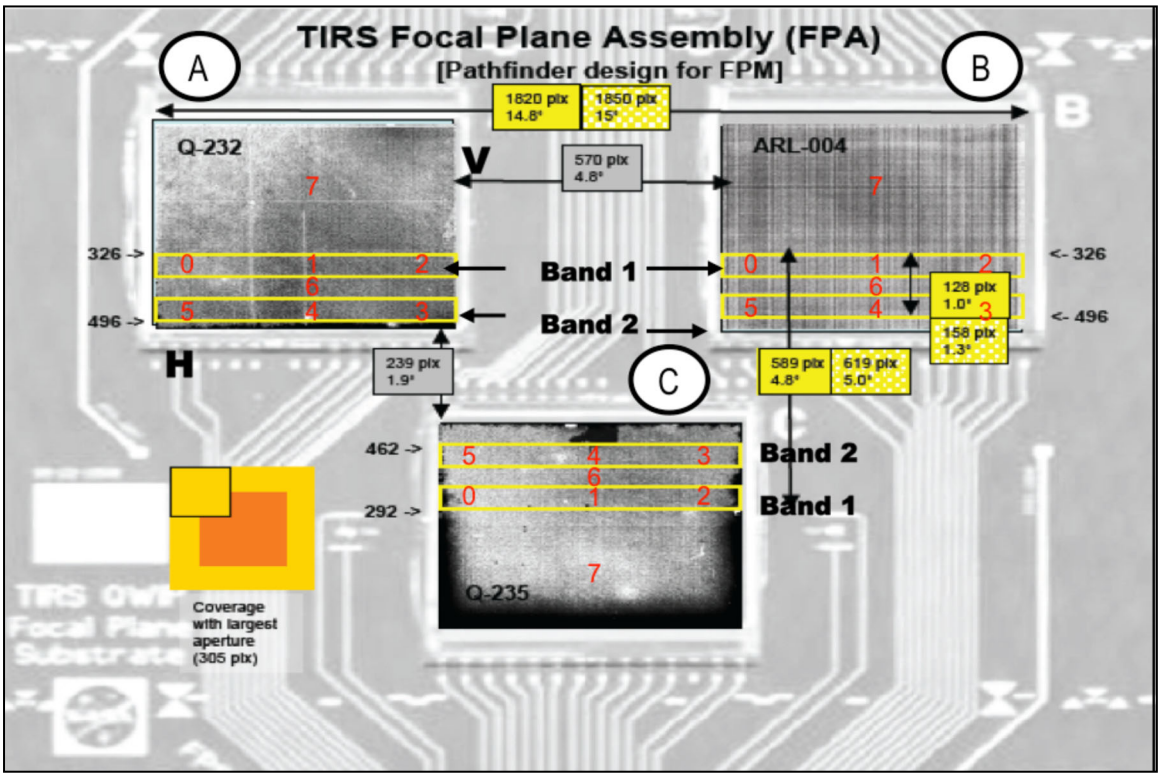

Figure 2-2. TIRS Focal Plane Assembly

#	Band	Center Wavelength (nm)	Center Wavelength Tolerance (±nm)	Minimum Lower Band Edge (nm)	Maximum Upper Band Edge (nm)
10	Thermal 1	10800	200	10600	11190
11	Thermal 2	12000	200	11500	12510

Table 2-2. Spectral Ranges of TIRS Bands

Section 3 Process Flows

To perform the characterization, calibration, and correction functions, described in the Landsat Data Continuity Mission Government Calibration and Validation Plan and Landsat 9 Government Calibration and Validation Plan, each of the algorithms (described in Section 4) are linked together in a processing flow. This processing flow is separated into OLI geometry, OLI radiometry, TIRS geometry, and TIRS radiometry.

3.1 OLI Geometry

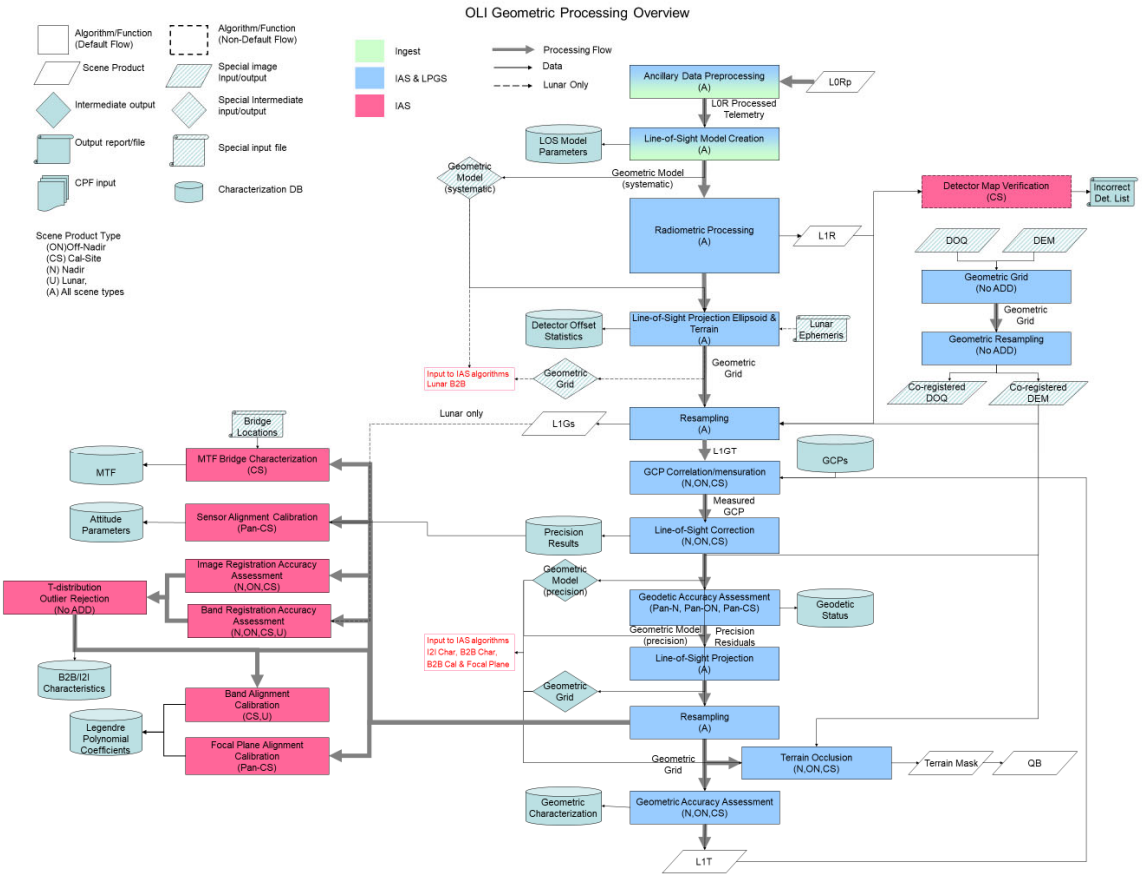

***Figure** 3-1. OLI Geometry Processing Flow Diagram*

3.2 OLI Radiometry

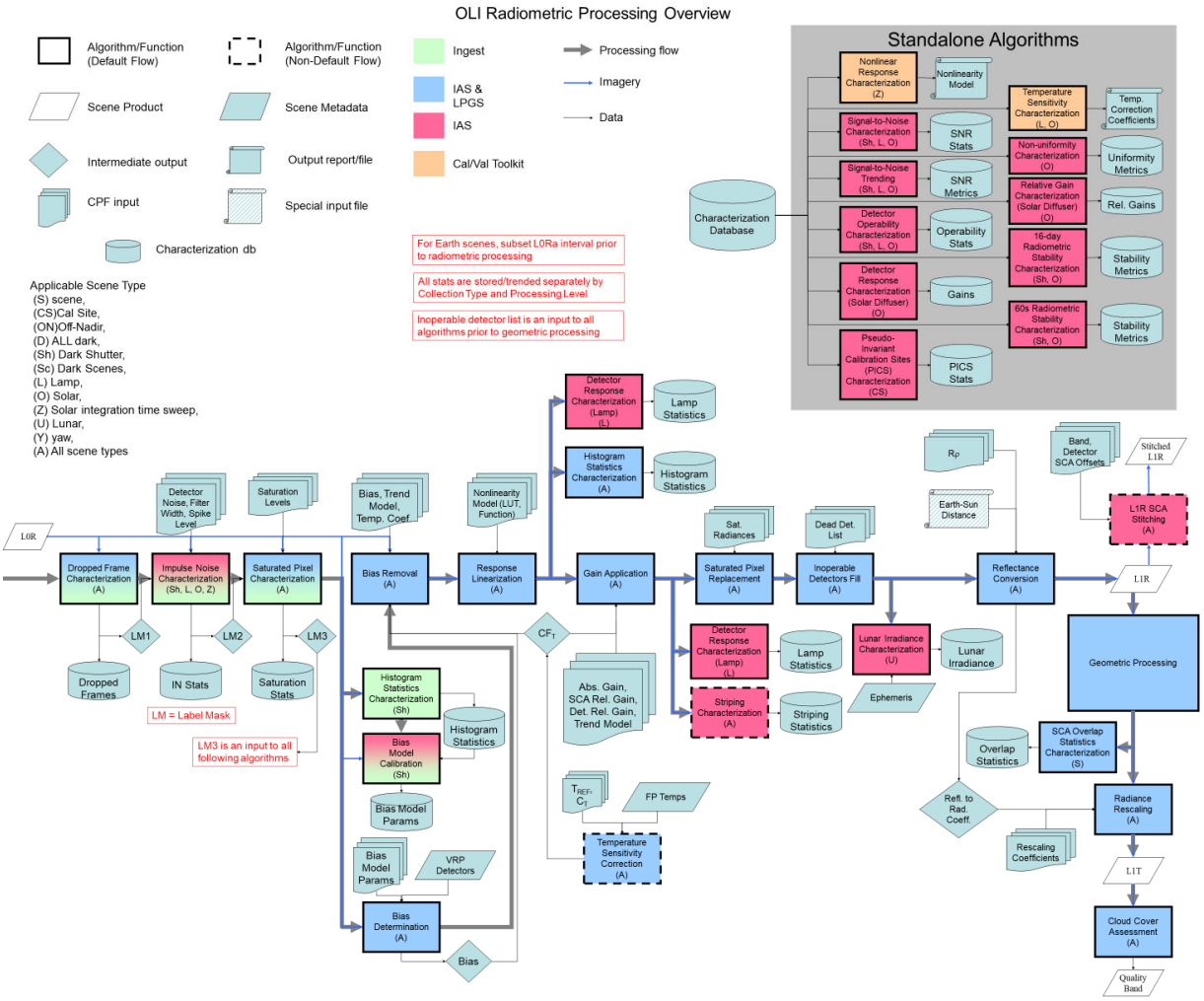

Figure** 3-2. **OLI Radiometry Processing Flow Diagram

3.3 TIRS Geometry

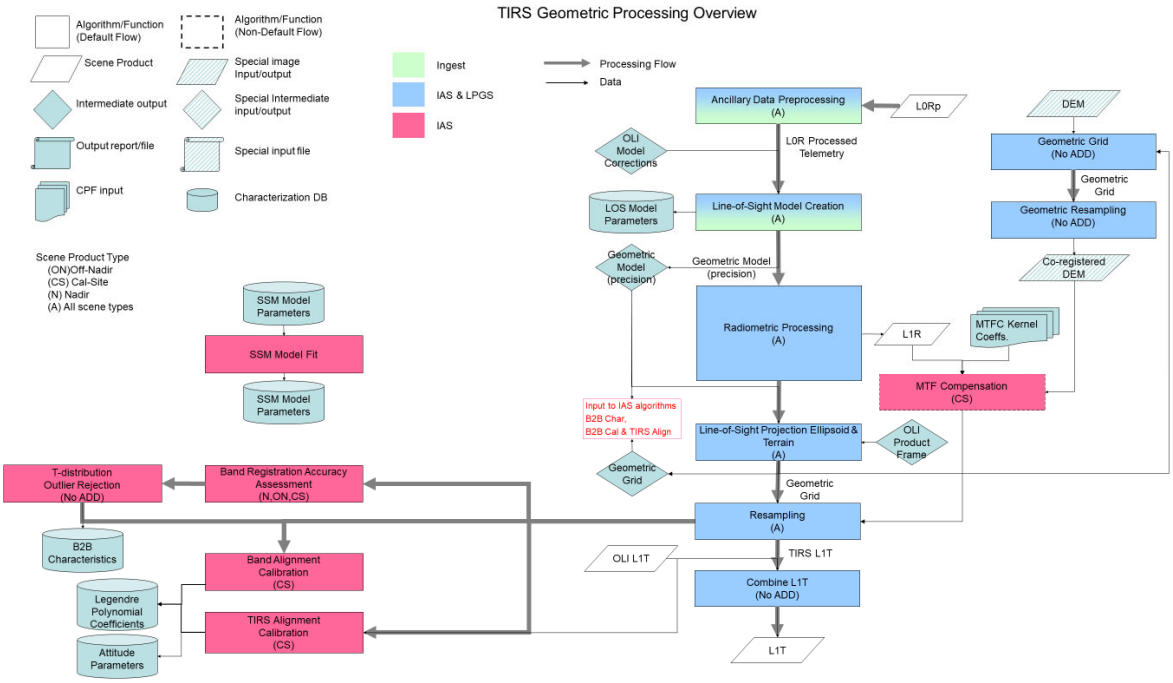

***Figure** 3-3. TIRS Geometry Processing Flow Diagram*

3.4 TIRS Radiometry

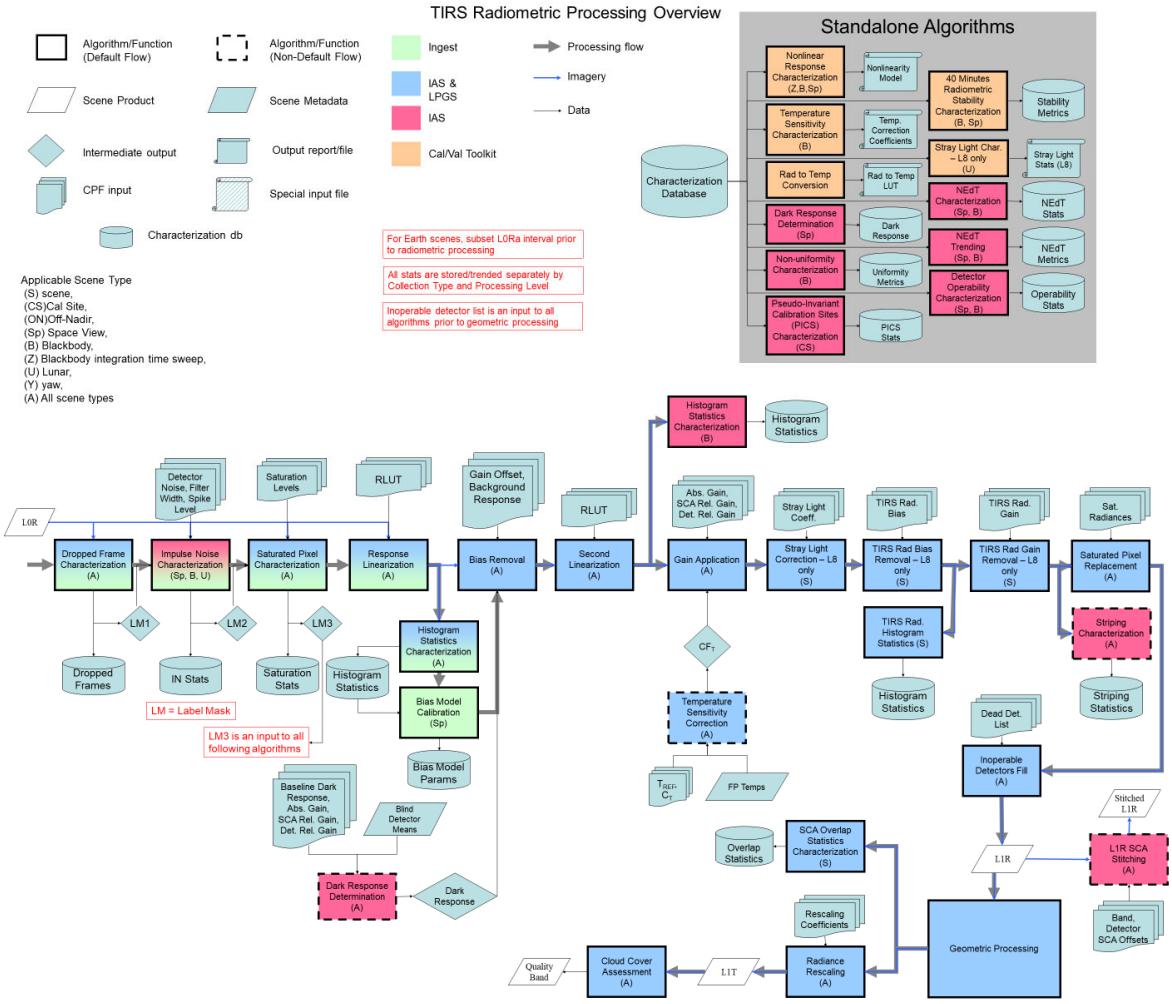

Figure** 3**-4. TIRS Radiometry Processing Flow Diagram

Section 4 Algorithms

4.1 Common Geometry Algorithms

4.1.1 Coordinate Systems

4.1.1.1 Coordinate System Definitions

The L8/9 IAS geometric algorithms use ten coordinate systems. These coordinate systems are referred to frequently in the remainder of this document and are briefly defined here to provide context for the subsequent discussion. They are presented in the order in which they would be used to transform a detector and sample time into a ground position.

1. OLI Instrument Line-of-Sight (LOS) Coordinate System

The OLI LOS coordinate system is used to define the band and detector pointing directions relative to the instrument axes. These pointing directions are used to construct LOS vectors for individual detector samples. This coordinate system is defined so that the Z-axis is parallel to the telescope boresight axis and is positive toward the OLI aperture. The origin is where this axis intersects the OLI focal plane. The X-axis is parallel to the along-track direction, with the positive direction toward the leading, odd numbered, SCAs (see Figure 4-1). The Y-axis is in the across-track direction, with the positive direction toward SCA01. This definition makes the OLI coordinate system nominally parallel to the spacecraft coordinate system, with the difference being due to residual misalignment between the OLI and the spacecraft body.

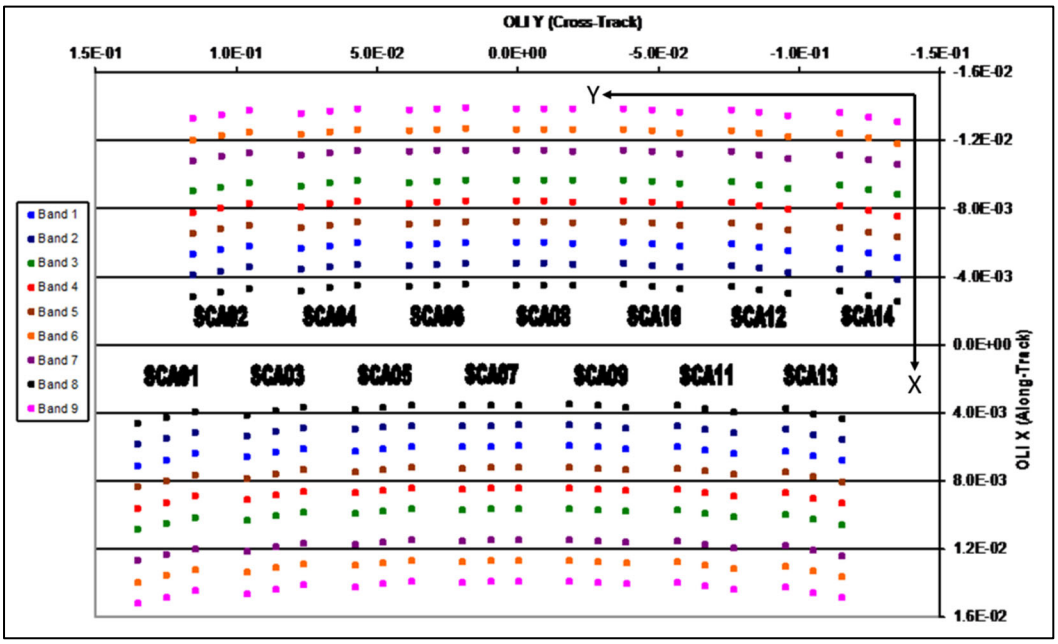

Figure 4-1. OLI Line-of-Sight Coordinate System

2. TIRS Instrument Coordinate System

   The orientations of the TIRS detector LOS directions and of the TIRS Scene Select Mirror (SSM) are both defined within the TIRS instrument coordinate system. TIRS LOS coordinates define the band and detector-pointing directions relative to the instrument axes. These pointing directions are used to construct LOS vectors for individual detector samples. These vectors are reflected off the SSM to direct them out the TIRS aperture for Earth viewing. The TIRS LOS model is formulated so that the effect of a nominally pointed SSM is included in the definition of the detector lines-of-sight, with departures from nominal SSM pointing causing perturbations to these lines-of-sight. This formulation allows TIRS LOS construction to be very similar to OLI, and is described in detail below, in the TIRS Line-of-Sight Model Creation algorithm (see 4.3.1).

   The TIRS coordinate system is defined so that the Z-axis is parallel to the TIRS boresight axis and is positive toward the TIRS aperture. The origin is where this axis intersects the TIRS focal plane. The X-axis is parallel to the along-track direction, with the positive direction toward the leading SCA, SCA02 (see Figure 4-2). The Y-axis is in the across-track direction, with the positive direction toward SCA03. This definition makes the TIRS coordinate system nominally parallel to the spacecraft coordinate system, with the difference being due to residual misalignment between the TIRS and the spacecraft body.

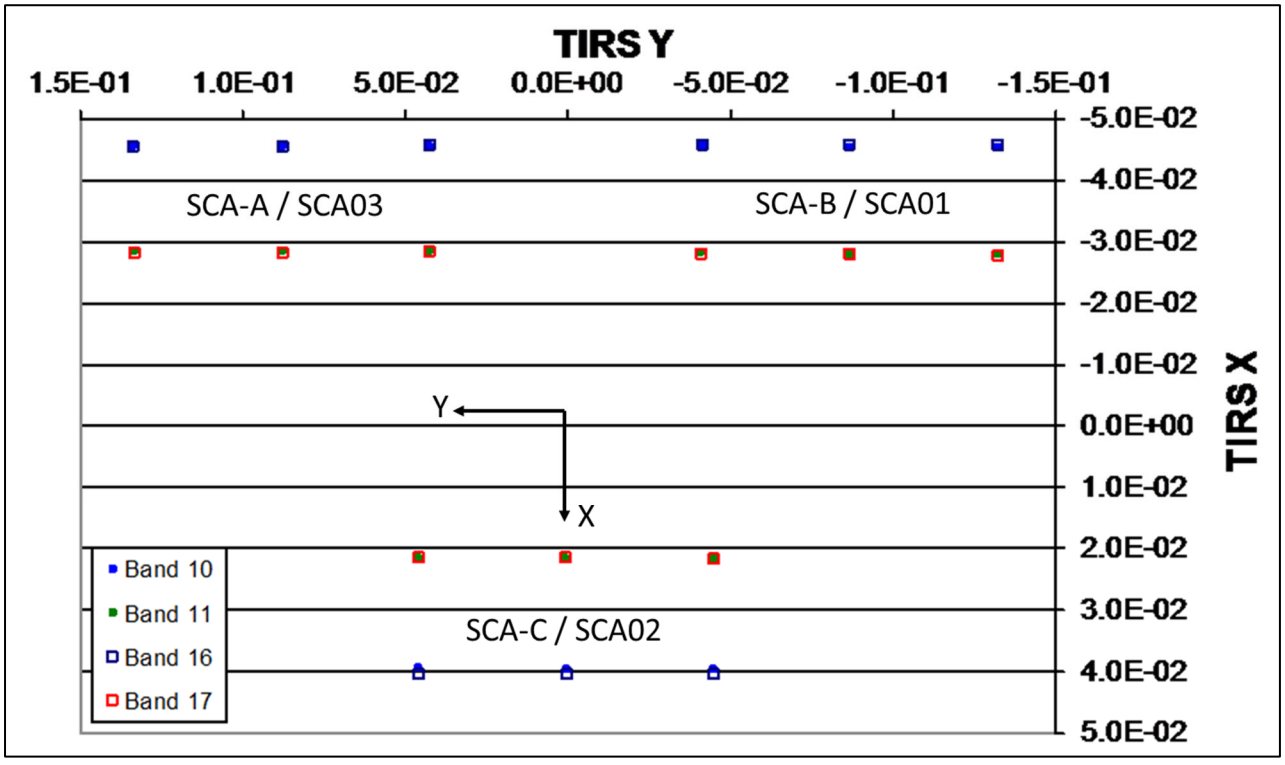

Figure 4-2. TIRS Line-of-Sight Coordinates

3. Spacecraft Coordinate System

   The spacecraft coordinate system is the spacecraft-body-fixed coordinate system used to relate the locations and orientations of the various spacecraft components to one another and to the OLI and TIRS instruments. It is defined with the +Z axis in the Earth-facing direction, the +X axis in the nominal direction of flight, and the +Y axis toward the cold side of the spacecraft (opposite the solar array). This coordinate system is useful during observatory integration and prelaunch test, where it is used to determine the prelaunch positions and alignments of the attitude control sensors (star trackers and Space Inertial Reference Unit (SIRU)) and instrument payloads (OLI and TIRS). The spacecraft coordinate system is nominally the same as the navigation reference system (see below) used for spacecraft attitude determination and control. However, for reasons explained below, these two coordinate systems are treated separately.

4. Navigation Reference Coordinate System

   The navigation reference frame (a.k.a., the attitude control system reference) is the spacecraft-body-fixed coordinate system used for spacecraft attitude determination and control. The coordinate axes are defined by the spacecraft Attitude Control System (ACS), which attempts to keep the navigation reference frame aligned with the (yaw-steered) orbital coordinate system (for nominal nadir pointing) so that the OLI and TIRS boresight axes are always pointing toward the center of the Earth. The orientation of this coordinate system relative to the inertial coordinate system is captured in spacecraft attitude data.

   Ideally, the navigation reference frame is the same as the spacecraft coordinate system. In practice, the navigation frame is based on the orientation of the absolute attitude sensor (i.e., star tracker) being used for attitude determination. Any errors in the orientation knowledge for this tracker with respect to the spacecraft body frame will lead to differences between the spacecraft and navigation coordinate systems. This becomes important if the absolute attitude sensor changes (for example, by switching from the primary to the redundant star tracker during on-orbit operations). Such an event would effectively redefine the navigation frame to be based on the redundant tracker, with the difference between the spacecraft and navigation frames now resulting from redundant tracker alignment knowledge errors, rather than from primary tracker alignment knowledge errors. This redefinition would require updates to the on-orbit instrument-to-ACS alignment calibrations. Therefore, the spacecraft and navigation reference coordinate systems are different because the spacecraft coordinate system is fixed but the navigation reference can change.

5. SIRU Coordinate System

   The spacecraft orientation rate data provided by the spacecraft attitude control system's inertial measurement unit are referenced to the SIRU coordinate system. The SIRU consists of four rotation-sensitive axes. This configuration provides redundancy to protect against the failure of any one axis. The four SIRU axis directions are determined relative to the SIRU coordinate system, the orientation of which is itself measured relative to the spacecraft coordinate system, both prelaunch and on-orbit, as part of the ACS calibration procedure. The IAS uses this alignment transformation to convert the SIRU data contained in the L8/9 spacecraft ancillary data to the navigation reference coordinate system for blending with the ACS quaternions.

6. Orbital Coordinate System

   The orbital coordinate system is centered at the spacecraft, and its orientation is based on the spacecraft position in inertial space (see Figure 4-3). The origin is the spacecraft's center of mass, with the Z-axis pointing from the spacecraft's center of mass to the Earth's center of mass. The Y-axis is the normalized cross-product of the Z-axis and the instantaneous (inertial) velocity vector, and corresponds to the negative of the instantaneous angular momentum vector direction. The X-axis is the cross-product of the Y and Z-axes. The orbital coordinate system is used to convert spacecraft attitude, expressed as Earth-Centered Inertial (ECI) quaternions, to roll-pitch-yaw Euler angles.

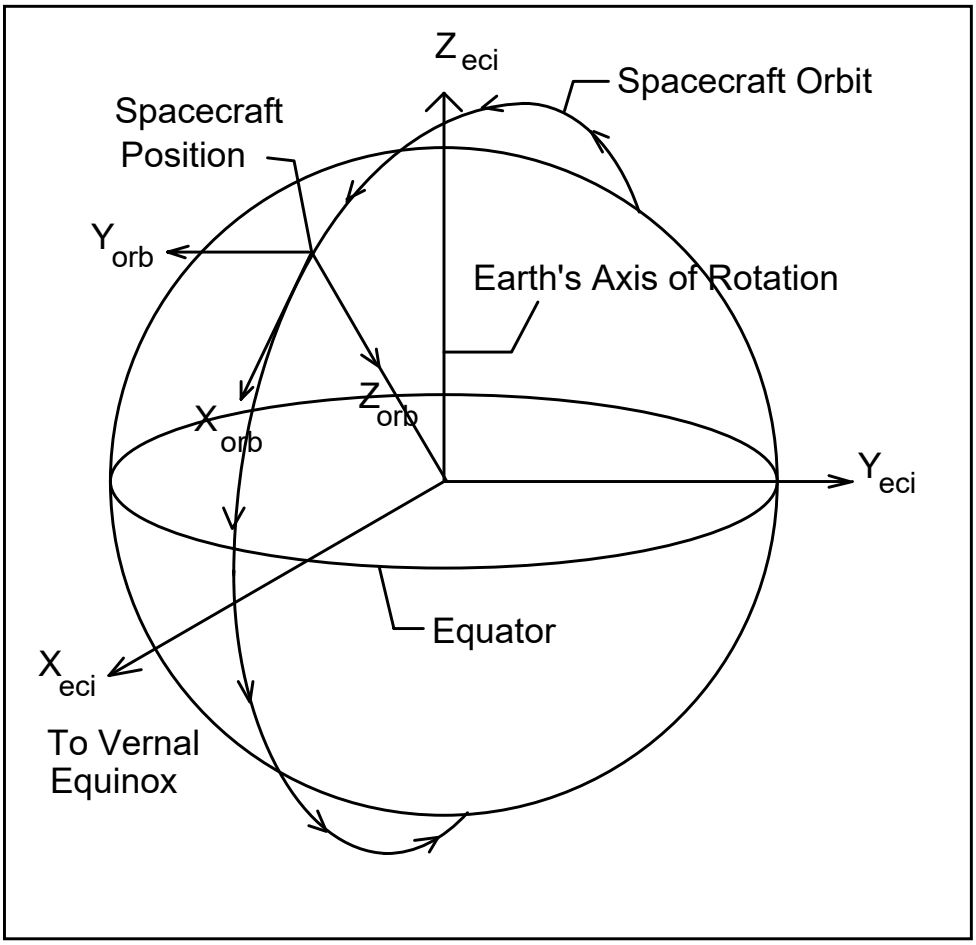

Figure 4-3. Orbital Coordinate System

7. ECI J2000 Coordinate System

   The ECI coordinate system of epoch J2000 is space-fixed with its origin at the Earth's center of mass (see Figure 4-4). The Z-axis corresponds to the mean north celestial pole of epoch J2000.0. The X-axis is based on the mean vernal equinox of epoch J2000.0. The Y-axis is the cross-product of the Z and X axes. The Explanatory Supplement to the Astronomical Almanac published by the U.S. Naval Observatory describes this coordinate system in detail. Data in the ECI coordinate system are present in the L8/9 spacecraft ancillary data in the form of attitude quaternions that relate the navigation frame to the ECI J2000 coordinate system.

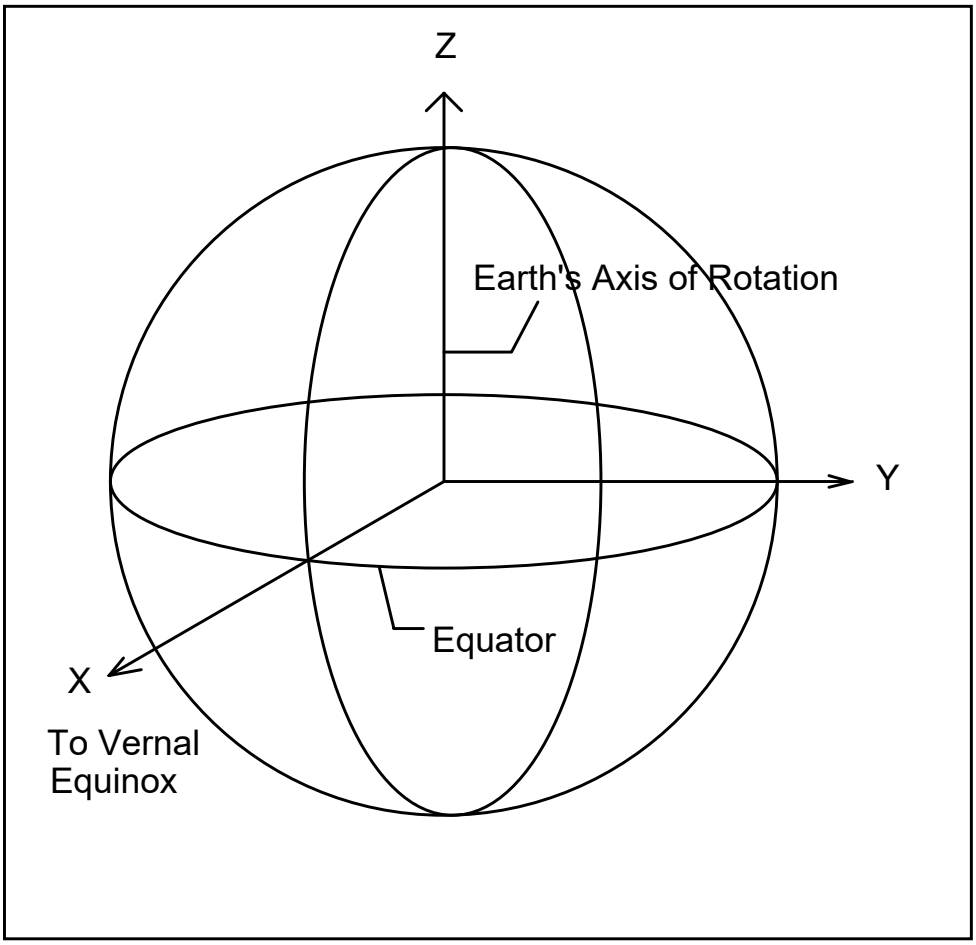

Figure 4-4. Earth-Centered Inertial (ECI) Coordinate System

8. ECEF Coordinate System

   The Earth-Centered Earth-Fixed (ECEF) coordinate system is Earth-fixed, with its origin at the Earth's center of mass (see Figure 4-5). It corresponds to the U.S. Department of Defense World Geodetic System 1984 (WGS84) geocentric reference system which, in its modern realizations, is aligned with the International Terrestrial Reference Frame (ITRF) defined by the International Earth Rotation Service (IERS). WGS84 is described in National Geospatial-Intelligence Agency (NGA) Standardization Document: Department of Defense World Geodetic System 1984: Its Definition and Relationships with Local Geodetic Systems, NGA.STND.0036_1.0.0_WGS84.

   Since both L8 and L9 use the Global Positioning System (GPS) for time and position determination, both are based on the realization of WGS84 implemented by the GPS. As of this writing, the current WGS84 realization is G1762 which was implemented by GPS on October 16, 2013. This WGS84 realization is consistent with ITRF2008 to the sub-centimeter level. The more recent ITRF2014 differs from ITRF2008 by only a few

millimeters. The ground control points used to register L8/L9 products to the Global Land Survey (GLS) reference were originally established circa 2000 through an image triangulation procedure that included WGS84/GPS reference points which would have been based upon an earlier realization of WGS84. Despite the difference in WGS84 realizations, the accuracy of the GLS ground control (multiple meters to tens of meters) is such that distinctions at the level of WGS84 or ITRF realizations are not meaningful. As the WGS84 reference frame evolves over time, the changes will effectively introduce an additional small error term into the GLS control framework.

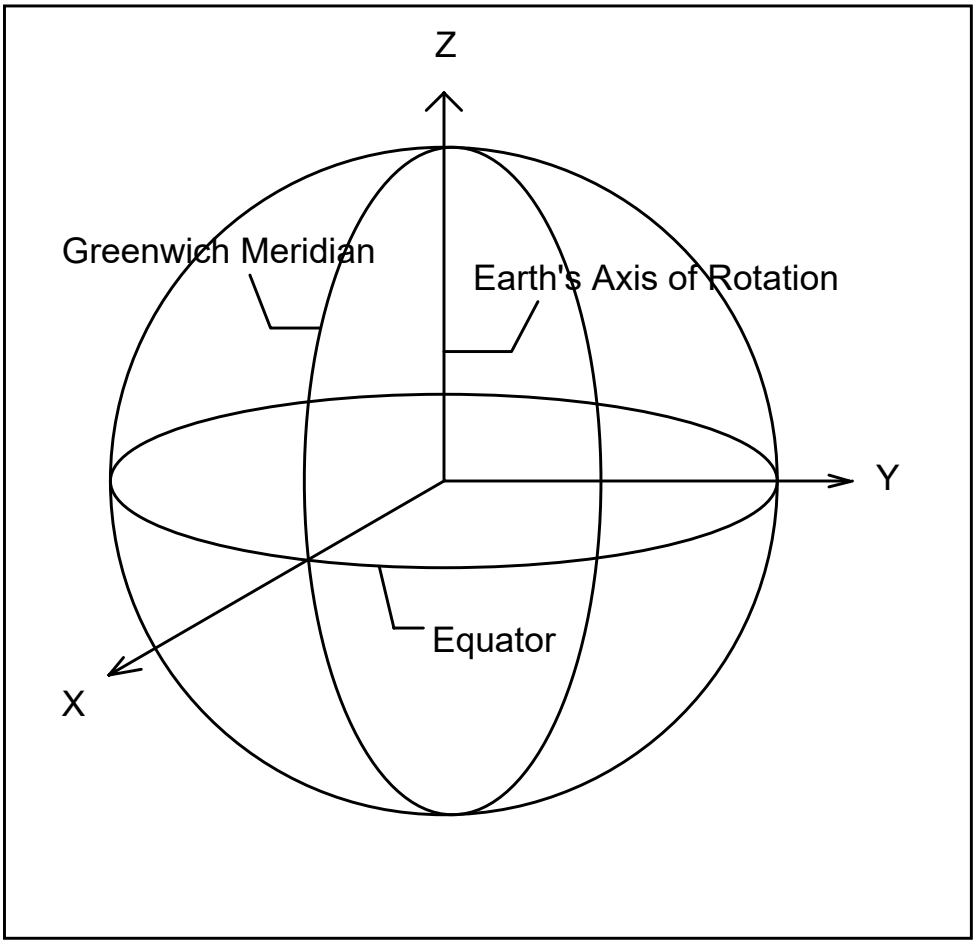

Figure 4-5. Earth-Centered Earth-Fixed (ECEF) Coordinate System

9. Geodetic Coordinate System

   The geodetic coordinate system is based on the WGS84 reference frame, with coordinates expressed in latitude, longitude, and height above the reference Earth ellipsoid (see Figure 4-6). No ellipsoid is required by the definition of the ECEF coordinate system, but the geodetic coordinate system depends on the selection of an Earth ellipsoid. Latitude is the angle between the ellipsoid normal and its projection onto the equator,

while longitude is the angle between the local meridian and the Greenwich meridian. The scene center and scene corner coordinates in the Level 0 Reformatted (L0R) product metadata are expressed in the geodetic coordinate system.

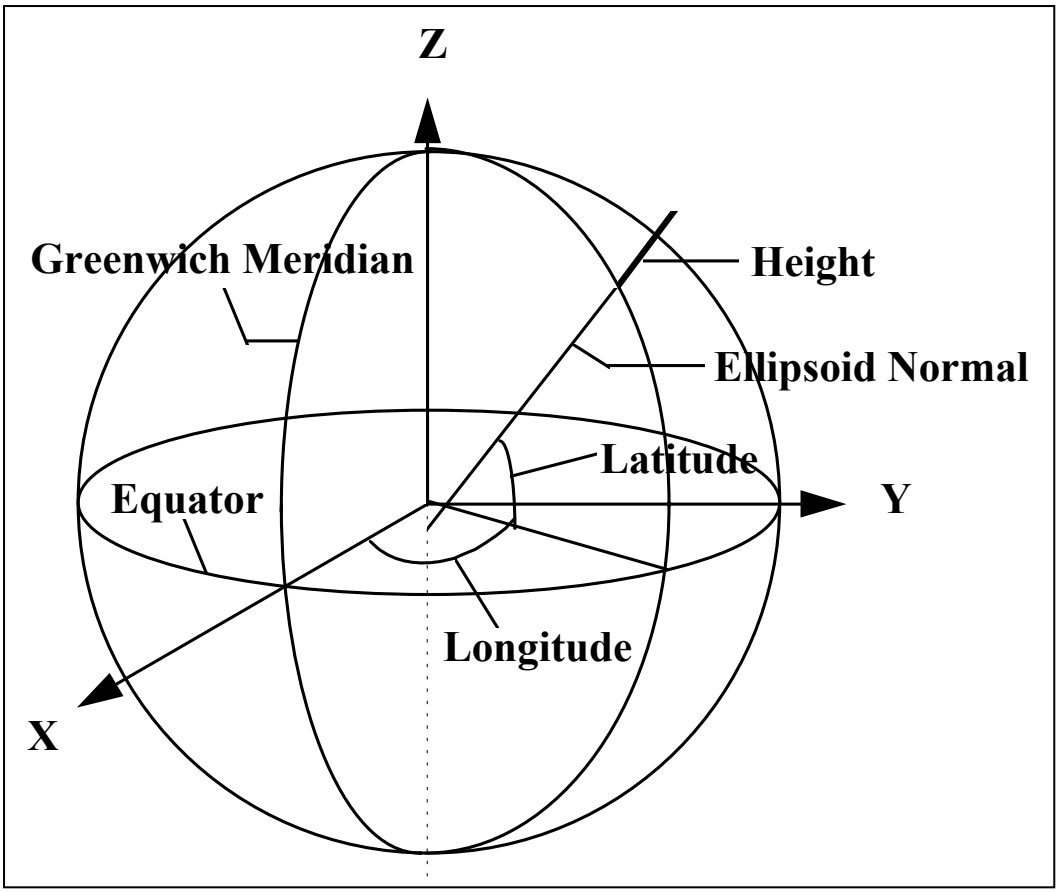

Figure 4-6. Geodetic Coordinate System

10. Map Projection Coordinate System

    Level 1 products are generated with respect to a map projection coordinate system, such as the Universal Transverse Mercator (UTM), which provides a mapping from latitude and longitude to a plane coordinate system that approximates a Cartesian coordinate system for a portion of the Earth's surface. It is used for convenience as a method of providing digital image data in an Earth-referenced grid that is compatible with other ground-referenced datasets. Although the map projection coordinate system is only an approximation of a true local Cartesian coordinate system at the Earth's surface, the mathematical relationship between the map projection and geodetic coordinate systems is defined precisely and unambiguously.

4.1.1.2 Coordinate Transformations

Eight key transformations relate the ten coordinate systems used by the IAS geometric algorithms. These transformations are referred to frequently in the remainder of this document and are defined here. They are in the logical order in which a detector and sample number would be transformed into a ground position.

1. OLI-to-Navigation Reference Transformation

   The OLI instrument alignment matrix describes the relationship between the OLI instrument and navigation reference coordinate systems. The transformation from sensor coordinates to navigation reference coordinates is a three-dimensional rotation, implemented as a matrix multiplication, and an offset to account for the distance between the ACS reference and the instrument aperture. This spacecraft center of mass-to-sensor offset is measured prelaunch and is not expected to be updated on-orbit. The ACS-to-OLI transformation matrix is initially defined as a static (non-time varying) rotation, with improved estimates provided post-launch. Subsequent analysis may detect repeatable variations with time, which can be effectively modeled, making this a (slowly) time-varying transformation. The nominal rotation matrix is the identity matrix.

2. TIRS-to-Navigation Reference Transformation

   The TIRS instrument alignment matrix describes the relationship between the TIRS instrument and navigation reference coordinate systems. Like the OLI, the transformation from sensor coordinates to navigation reference coordinates is a three-dimensional rotation, implemented as a matrix multiplication, and an offset to account for the distance between the ACS reference and the instrument aperture. This spacecraft center of mass-to-sensor offset is measured prelaunch and is not expected to be updated on-orbit. The ACS-to-TIRS transformation matrix is measured directly prelaunch. Post-launch, improved estimates will be provided by estimating the OLI-to-TIRS alignment and combining that with the ACS-to-OLI alignment previously described. Note that any TIRS pointing offsets that are due to errors in SSM alignment knowledge will be attributed to the overall ACS-to-TIRS alignment by the on-orbit calibration unless the SSM is intentionally pointed off-nadir to allow the SSM rotation axis to be calibrated. The nominal ACS-to-TIRS rotation matrix is the identity matrix.

3. SIRU-to-Navigation Reference Transformation

   The SIRU coordinate system is related to the navigation reference coordinate system by the SIRU alignment matrix, which captures the orientation of the SIRU axes with respect to the navigation base. This transformation is applied to the SIRU measurements present in the

spacecraft ancillary data prior to their integration with the ACS quaternions. The SIRU alignment is measured pre-flight, and is nominally oriented with a 45-degree rotation about the X-axis, relative to the spacecraft/navigation reference coordinate system.

4. Navigation Reference-to-Orbital Transformation

   The spacecraft attitude defines the relationship between the navigation reference and orbital coordinate systems. This transformation is a three-dimensional rotation matrix, with the components of the rotation matrix being functions of the spacecraft roll, pitch, and yaw attitude angles. The nature of the functions of roll, pitch, and yaw depends on the exact definition of these angles. The conventions adopted in the L8/9 model are described below in the Ancillary Data Preprocessing algorithm (4.1.4). Since the spacecraft attitude is constantly changing, this transformation is time varying. The nominal rotation matrix consists of a latitude-dependent rotation about the Z-axis (yaw). This “yaw-steering” is designed to compensate for the effects of Earth rotation as spacecraft motion passes the OLI and TIRS detector arrays over the Earth’s surface.

5. Orbital-to-ECI Transformation

   The relationship between the orbital and ECI coordinate systems is based on the spacecraft's instantaneous ECI position and velocity vectors. The rotation matrix to convert from orbital to ECI can be constructed by forming the orbital coordinate system axes in ECI coordinates:

   $\underline{P}$ = spacecraft position vector in ECI
   $\underline{V}$ = spacecraft velocity vector in ECI
   $T_{eci/orb}$ = rotation matrix from orbital to ECI

   $\underline{b}_3 = -\underline{p} / |\underline{p}|$ (nadir vector direction)
   $\underline{b}_2 = (\underline{b}_3 \times \underline{v}) / |\underline{b}_3 \times \underline{v}|$ (negative of angular momentum vector direction)
   $\underline{b}_1 = \underline{b}_2 \times \underline{b}_3$ (circular velocity vector direction)
   $T_{eci/orb} = [\ \underline{b}_1\ \ \underline{b}_2\ \ \underline{b}_3\ ]$

6. ECI-to-ECEF Transformation

   The transformation from ECI-to-ECEF coordinates is a time-varying rotation due primarily to the Earth’s rotation, but it also contains more slowly varying terms for precession, astronomic nutation, and polar wander. The ECI-to-ECEF rotation matrix is expressed as a composite of these transformations:

   $T_{ecr/eci} = A\ B\ C\ D$

A = polar motion
B = sidereal time
C = astronomic nutation
D = precession

Each of these transformation terms is described in more detail below in the Ancillary Data Preprocessing algorithm (4.1.4). Note that L8/9 uses the precession, nutation, and sidereal time definitions from the IAU resolutions of 1997-2000, as described in U.S. Naval Observatory Circular Number 179 dated October 20, 2005.

7. ECEF-to-Geodetic Transformation

   The relationship between ECEF and geodetic coordinates can be expressed simply in its direct form:

$$e^2 = 1 - b^2 / a^2$$
$$N = a / (1 - e^2 \sin^2(lat))^{1/2}$$
$$X = (N + h) \cos(lat) \cos(lon)$$
$$Y = (N + h) \cos(lat) \sin(lon)$$
$$Z = (N (1 - e^2) + h) \sin(lat)$$

where:

X, Y, Z = ECEF coordinates
lat, *lon*, *h* = geodetic coordinates
N = ellipsoid radius of curvature in the prime vertical
e^2 = ellipsoid eccentricity squared
a, b = ellipsoid semi-major and semi-minor axes

The closed-form solution for the general inverse problem (the problem of interest here) involves the solution of a quartic (fourth order) equation, and is not typically used in practice. Instead, an iterative solution is used for latitude and height for points that do not lie on the ellipsoid surface.

8. Geodetic-to-Map Projection Transformation

   The transformation from geodetic coordinates to the output map projection depends on the type of projection selected. John P. Snyder's Map Projections – A Working Manual, USGS Professional Paper 1395 gives the mathematics for the forward and inverse transformations for the UTM, Albers Equal Area Conic, Lambert Conformal Conic, Transverse Mercator, Oblique Mercator, Polyconic, Polar Stereo Graphic, and the Space Oblique Mercator (SOM) projections, among others.

4.1.2 Time Systems

Five time systems are of primary interest for the IAS geometric algorithms: Terrestrial Time (TT), International Atomic Time (Temps Atomique International [TAI]), Coordinated Universal Time (UTC), Universal Time—Corrected for polar motion (UT1), and Spacecraft Time (the readout of the spacecraft clock, derived from GPS time). Terrestrial time is the astronomic time system used for planetary and stellar ephemerides (e.g., star catalogs) and to model Earth precession and nutation. It is thus used in the conversion from Earth-centered inertial (ECI) of epoch J2000 coordinates to ECI true-of-date coordinates. Spacecraft Time is the time system used for the spacecraft time codes found in the Level 0R ancillary data (including image time codes). UTC is the standard reference for civil timekeeping. UTC is adjusted periodically by whole leap seconds to keep it within 0.9 seconds of UT1. UT1 is based on the actual rotation of the Earth and is needed to provide the transformation from stellar-referenced inertial coordinates (ECI true-of-date) to terrestrial-referenced Earth-centered Earth-fixed coordinates (ECEF). TAI provides a uniform, continuous time stream that is not interrupted by leap seconds or other periodic adjustments. It provides a consistent reference for resolving ambiguities arising from the insertion of leap seconds into UTC, which can lead to consecutive seconds with UTC times that appear to be the same. Spacecraft time is based on GPS time, which is, itself, a fixed offset from TAI. The Explanatory Supplement to the Astronomical Almanac, mentioned previously, describes these and a variety of other time systems and their relationships. The following text describes the significance of each of these time systems, with respect to the IAS geometric algorithms.

1. Terrestrial Time

   Epoch J2000 occurred at January 1, 2000 12:00:00 Barycentric Dynamical Time (TDB). Terrestrial Dynamical Time (TDT), known simply as Terrestrial Time (TT) since the IAU resolutions of 1991, is the time scale used for apparent geocentric ephemerides. At the time of the J2000 epoch, TT differed from TDB by approximately 73 microseconds (ref. Explanatory Supplement to the Astronomical Almanac). This small difference is ignored in the definition above, and the J2000 epoch is effectively taken to be January 1, 2000, 12:00:00 TT. The difference between TDB and TT is due to relativistic effects and is a periodic function containing terms as large as 1-2 milliseconds. This difference is accounted for by the USNO NOVAS software in its precession and nutation computations, but this is transparent to the time and coordinate system conversion logic that invokes the NOVAS functions.

   Epoch J2000.0 TT is now considered to be the “standard epoch” for modern astrometric reference data (Ref. USNO Circular 179). TT is defined to be TAI + 32.184 seconds, so the J2000 epoch is 12:00:00 TT = 11:59:27.816 TAI + 32.184 sec. Furthermore, at the time of the J2000 epoch, TAI and UTC differed by 32 accumulated leap seconds, so 11:59:27.816 TAI = 11:58:55.816 UTC + 32.000 leap sec. Note that the relationship between TT and TAI is fixed, but the relationship between TT and UTC changes over time,

with the offset increasing by one second each time a new leap second is declared. As of the date of this document, five additional leap seconds have been declared since the J2000 epoch: in January 2006, January 2009, July 2012, June 2015, and January 2017.

2. Spacecraft Time

In accordance with the Landsat Data Continuity Mission Spacecraft to Ground Interface Control Document (70-P58230P, Rev C), the L8 spacecraft clock reports time as TAI seconds since the spacecraft epoch, defined to coincide with the J2000 epoch:

January 1, 2000, 11:59:27.816 TAI.

Which is the same as the following:

January 1, 2000, 11:58:55.816 UTC.

Although L8 and L9 both use GPS as the time reference source, they use different GPS receivers. The L9 receiver is constrained to issue one-pulse-per-second (PPS) time marks only on whole GPS second boundaries. The ACS flight software operates on a 50 Hz cycle synchronized to the 1 PPS signals, with particular events (e.g., polling attitude sensors, computing attitude updates) occurring on predefined 50 Hz clock ticks. Using a clock epoch that is not aligned with GPS second boundaries would complicate the management of critical timing relationships in the flight software. As a result, the decision was made (by Northrup Grumman) for L9 that it would be simpler to define the spacecraft clock epoch as a whole number of seconds offset relative to TAI. This makes the offset between the spacecraft clock and the GPS system a whole number of seconds since the GPS epoch is aligned with TAI. The L9 spacecraft clock thus reports time as TAI seconds since the spacecraft epoch, defined as follows:

January 1, 2000, 11:59:28.000 TAI.

Which is the same as the following:

January 1, 2000, 11:58:56.000 UTC.

The L9 spacecraft clock epoch is thus 0.184 seconds later than the L8 spacecraft clock epoch. This “round off” offset has to be taken into account when converting L9 spacecraft time codes into seconds from J2000.0 TT for purposes of ECI to ECEF conversion. This clock epoch conversion will be performed during Ingest processing, taking advantage of the fact that the L0R data format already contains separate fields for “L0R Time” (as days and seconds of day since the J2000 epoch), and for “Original Spacecraft Time”

(as days, milliseconds, and microseconds of day), so that both the corrected (J2000) and uncorrected (clock epoch) times are available in the L0R data.

The L8/9 flight software maintains the accuracy of the spacecraft clock, using time data from the on-board GPS receiver(s). The spacecraft clock is then used to time-tag the spacecraft ancillary data and to provide a timing reference for the OLI and TIRS instruments. As noted above, spacecraft time is used to define the times at which the flight software generates filtered attitude and ephemeris estimates based on the input GPS, star tracker, and SIRU data. These estimates are included in the spacecraft ancillary data stream for use by ground processing. Also included in the ancillary data are the raw SIRU measurements. Individual SIRU observations are time tagged using a clock/counter internal to the SIRU itself, but the SIRU ancillary data also includes SIRU time synch events that make it possible to relate the SIRU clock to spacecraft time.

The spacecraft clock also provides time synchronization signals to the OLI and TIRS instruments once per second. Both instruments use the 1 PPS signals to regulate their internal clocks, thereby registering the image time codes to spacecraft time. Note that any instrument clock rate errors will manifest as (small) step discontinuities in the image time codes, which correspond to the 1 PPS updates. The resulting time code irregularities are corrected when the OLI and TIRS geometric models are created, as described below in the OLI LOS Model Creation algorithm and the TIRS LOS Model Creation algorithm.

3. UTC

As mentioned above, UTC is maintained within 0.9 seconds of UT1 by the occasional insertion of leap seconds. A table of leap seconds relating UTC to TAI is maintained in the L8/9 CPF to support the spacecraft time to UTC conversion. To convert spacecraft time to UTC, the number of additional leap seconds declared since the spacecraft epoch are subtracted from the reported spacecraft seconds since epoch and the result is added to the UTC representation of the epoch presented above. Leap second information is available from the International Earth Rotation Service (IERS) in their Bulletin C publications.

4. UT1

UT1 represents time with respect to the actual rotation of the Earth, and is used by the IAS algorithms, which transform inertial ECI coordinates or lines of sight to Earth-fixed ECEF coordinates. Failure to account for the difference between UT1 and UTC in these algorithms can lead to ground position errors as large as 400 meters at the equator (assuming the maximum 0.9-second UT1-UTC difference). The UT1-UTC correction typically varies at the rate of approximately 2 milliseconds per day, corresponding to an Earth rotation error of about 1 meter. Thus, UT1-UTC corrections should be interpolated or

predicted to the actual image acquisition time to avoid introducing errors of this magnitude. The UT1-UTC offset, along with the polar wander Earth orientation parameters, can be obtained from IERS Bulletin B (for retrospective data) and Bulletin A (for predicted data). The L8/9 CPF also includes tables of the UT1-UTC and polar wander Earth orientation parameters.

5. TAI

Although the IAS algorithms do not operate directly in TAI, it underlies the definition of spacecraft time, as noted above. As such, it can be helpful to use TAI as a standard reference that can be related to UTC, using the CPF leap second file, and to spacecraft time, via the constant offset, to assist IAS operations staff in anomaly resolution.

4.1.3 Scene Framing Algorithm

4.1.3.1 Background/Introduction

The L8/9 scene framing algorithm uses the spacecraft ancillary data, preprocessed to perform scaling, coordinate conversion, and to repair errors, and the image timing information to determine the locations of scene centers within the interval. It then assigns Worldwide Reference System-2 (WRS-2) path-row coordinates to these scene centers for purposes of subsequent metadata generation.

The Landsat heritage scene framing algorithm will be used to frame nadir-pointing data and to determine the nadir path/row references for off-nadir acquisitions, but the L8/9 capability to point (roll) up to 15 degrees off-nadir will lead to data acquisitions that do not fall on the regular WRS-2 reference grid and will, in some cases, fall entirely outside of the heritage WRS-2 coverage area for acquisitions near the poles. These additional complications require adjustments to the heritage algorithm to address both the scene definition (i.e., where do we declare the scene centers) aspect of scene framing and the WRS-2 grid (path/row) assignment aspect of scene framing. This algorithm addresses those requirements.

The algorithm developed here separates the scene definition and WRS-2 labeling aspects of the scene-framing problem. It also uses different logic for high-latitude (polar region) and low-latitude (non-polar) acquisitions. At high latitudes, off-nadir acquisitions will be poorly aligned with and will sometimes fall outside of the WRS-2 grid, so the heritage (nadir) orbit-based approach to scene center time definition is used in those areas. Using the nadir path/row for scene and interval identification also ensures unique ids as well as consistency with planned acquisitions. Furthermore, to help in identifying coverage of off-nadir acquisitions, especially near the poles, *target* WRS-2 labeling is determined, and any imagery falling outside of the WRS-2 grid uses special target row numbering.

For non-polar regions, the guiding principle is to make even off-nadir scenes as consistent as possible in coverage with the nadir acquisitions of the same region by using a "row-based" approach to scene definition. Defining scene centers at the locations where the OLI[1] boresight crosses the latitudes that correspond to WRS-2 row centers, makes the off-nadir scene latitude bounds align with the nadir-viewing scenes from adjacent paths. This should lead to greater consistency in scene coverage and improve the interoperability of nadir and off-nadir data[2]. Therefore, for non-polar regions, scene path/row computation is based on the boresight LOS intersection location. The basic principle in this portion of the algorithm is to treat the boresight location as if it were a sub-satellite point. We can then compute a corresponding orbital central travel angle and apparent descending node based on the nominal Landsat orbital inclination. The actual orbit data are used to determine whether the scene is ascending or descending mode, and the central travel angle and descending node are adjusted accordingly. The central travel angle is used to derive the WRS-2 row and the descending node longitude yields the (fractional) path. Since the scene definition logic is defined by WRS-2 row crossings, for non-polar data, the off-nadir scene paths will typically be fractional and the row will be an integer, whereas in polar areas, the off-nadir scenes target WRS-2 paths and rows will be fractional. These fractional WRS-2 path/rows will have to be rounded to the nearest path/row that represents the data. Target WRS-2 coordinates of nadir-pointing scenes should be integers in both regions. In addition, if the scene center falls too far off the WRS-2 grid, special target row numbers will be assigned (880-88x for the North Pole and 990-99x for the South Pole).

It is important to note that in the metadata, the wrs_path/wrs_row values are always the nadir or orbital path/row of the satellite (and they are used for the scene ids) and the *target* path/row is the LOS path/row. In the non-polar regions, the target row and orbital wrs_row should be equal for nadir viewing imagery. The target path should be between wrs_path-1 and wrs_path+1 at low latitudes, but could vary by more at higher latitudes (approaching the polar regions). In the polar region, off-nadir target path/row will vary quite a bit from the orbital values.

4.1.3.2 Dependencies

The scene-framing algorithm assumes that ancillary data for the full imaging interval, with 8 seconds of extra ancillary data before and after the imagery, is available to provide the required geometric support data; that a CPF containing Earth orientation parameters and OLI & TIRS alignment and offset information is available; and that the image time codes are available.

At a minimum, four seconds of ancillary data are required before and after the imagery to construct consistent attitude and ephemeris time histories for the

[1] For any Thermal Infrared Sensor (TIRS) only Earth acquisitions, the TIRS boresight is used.
[2] In contrast, framing all off-nadir data based on the spacecraft position instead of the boresight location was rejected, as it would lead to off-nadir scenes exhibiting an along-track shift relative to the adjacent path nadir data.

dataset in order to achieve L8/9 geolocation accuracy requirements. An additional four seconds is added as an allowance for late starts/stops, bad/partial frames at the beginning/end, etc. In addition, not all of the instrument telemetry is actually updated in every ancillary data frame, so it takes multiple (up to 4) frames to guarantee a fresh sample.

For calibration collects (lamp, solar, shutter, black body, deep space, etc.), the geolocation framing is not done, and a minimum of two seconds of ancillary are expected before and after the imagery.

A check to ensure the imagery is fully covered by ancillary data should be done at a minimum for all intervals. To avoid potential framing errors, a check for at least 4 seconds of ancillary data before and after the imagery for Earth collects may be prudent. If there is insufficient ancillary data to cover the imagery or processing results in framing errors, the imagery may have to be trimmed to fit within the available ancillary in order to process the data. A tool would have to be written to do this, or Ingest updated to “trim” imagery to fit within the ancillary data.

4.1.3.2.1 Addendum to Image Trimming

In order to support mission data files from a Live Downlink (as ICs would receive), image “trimming” is being looked at as an Ingest enhancement. The top option being considered is to not actually remove the imagery, but to indicate an interval start/stop frame that fits within the ancillary data, where processing would begin/finish. In that way, no data are thrown away, and there could potentially be updates to “extrapolate” the extent of the ancillary such that sometime in the future this imagery could be processed.

4.1.3.3 Inputs

The scene-framing algorithm and its component sub-algorithms use the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs
ODL File (implementation)
Satellite Attributes
Spacecraft Clock Epoch TAI Time Reference
CPF
Earth orientation parameters (UT1-UTC, pole wander, leap seconds)
OLI/TIRS Focal Plane Parameters
OLI/TIRS Parameters
Attitude Parameters
Orbit Parameters
WRS-2 Scene average elevation look-up file
Preprocessed Ancillary Data
Attitude Data
Attitude data UTC epoch: Year, Day of Year, Seconds of Day

Algorithm Inputs
Time from epoch (one per sample, nominally 50 Hz) in seconds
ECI quaternion (vector: q1, q2, q3, scalar: q4) (one per sample)
ECEF quaternion (one per sample)
Body rate estimate (roll, pitch, yaw rate) (one per sample) in radians/second
Roll, pitch, yaw estimate (one per sample) in radians
Ephemeris Data
Ephemeris data UTC epoch: Year, Day of Year, Seconds of Day
Time from epoch (one per sample, nominally 1 Hz) in seconds
ECI position estimate (X, Y, Z) (one set per sample) in meters
ECI velocity estimate (Vx, Vy, Vz) (one set per sample) in meters/second
ECEF position estimate (X, Y, Z) (one set per sample) in meters
ECEF velocity estimate (Vx, Vy, Vz) (one set per sample) in meters/second
L0R Data Contents
OLI Image Time Codes (one per frame)
TIRS Image Time Codes (one per frame)
Instrument-specific minimum number of frames per full scene (O:7001,T:2801)
Instrument-specific minimum number of overlap frames per scene (O:1322 [756 FOV + 566 overlap], T:1080 [800FOV + 168 overlap+ 112 misalignment])
Number of polar region rows for nadir scene center calculations (6)
Minimum latitude for special target row numbering (82.61deg) [Highest latitude if the roll shifts the imagery ~50% off the nadir pointing ground coverage.]

Note: Both ECI and ECEF attitude and ephemeris data are specified as inputs, but the baseline algorithm makes use of only the ECEF versions.

4.1.3.4 Outputs

The following table contains the scene-framing algorithm outputs.

Scene Framing Data
Number of scenes in the interval
For each scene in the interval:
Scene center time: year, day of year, seconds of day UTC
Scene start time: year, day of year, seconds of day UTC
Scene stop time: year, day of year, seconds of day UTC
Scene corner information.
OLI start/center/stop frame numbers
TIRS start/center/stop frame numbers
Assigned orbital WRS-2 path
Assigned orbital WRS-2 row
Assigned target WRS-2 path
Assigned target WRS-2 row

4.1.3.4.1 Approach Overview

Scene Framing consists of the following:

1. Using the ephemeris data, determine the WRS-2 orbital (i.e., nadir) range – this is also called the “heritage method.”
2. Determine scene center times for each row at the focal plane boresight position (OLI: Boresight is located between SCA7 & 8, TIRS: Boresight is located in the middle of SCA A, B, & C). This is performed by three different methods, depending on the row identified:
    a. LOS latitude crossing for non-polar rows,
    b. Nadir position for Polar Region Rows (within +/-6 rows of the poles),
    c. Zero Z-Velocity time for Polar Rows (246 or 122).
3. Determine each instrument’s scene center frame number based on the center time, from which the start and stop frame numbers and times are determined using each instrument’s minimum frame counts (given above).
4. Check and adjust scene start/stop times to ensure adequate overlap. This is mainly needed for off-nadir collects that transition to/from the polar region rows.
5. From each scene’s center coordinates, determine the closest Target Path/Row. This helps determine the closest WRS-2 coverage for off-nadir imaging. In addition, for extremely high latitude, special target row numbers are used (88x and 99x) to help identify imagery viewed “off the WRS-2 grid” around the poles.
6. Scene metadata (corner points, sun azimuth/elevation, etc.) are then completed.

4.1.3.5 Scene Framing Definitions and Guidelines

Note: The functionality described in this algorithm relies on geometric processing capabilities described in other OLI/TIRS algorithm description documents. These algorithms will be referenced as necessary.

4.1.3.5.1 Scene Sizes

Full scene declaration is based on a minimum number of frames (OLI: 7001, TIRS:2801); anything less is considered a partial scene. See Scene Sizing Background (4.1.3.8.1) for an overview of the fields-of-view and calculations to define the number of OLI and TIRS frames in a full WRS-2 scene. The L8/9 imaging instruments (OLI and TIRS) are pushbroom instruments with significantly large fields-of-view in the along-track direction. In addition, both instruments have redundant detectors that can be selected for active imaging. Since it is desirable to have a constant minimum frame size for Level-0 scenes to be considered “full,” the minimum scene frame size for each instrument is set large enough to always ensure enough coverage to produce full Level-1 scenes, which are approximately 180km in length. There is also a one-second tolerance on the boresight alignment between the two instruments.

The standard length of a full Level-0 OLI scene is given by the following:

OLI Sample Rate	4.236	Milliseconds / frame
OLI FOV SCA Staggering	1.7	Degrees (along-track) The angle required for the leading and trailing imaging bands of the SCAs to cover a point on the ground relative to the center of the focal plane. See Scene Sizing Background (4.1.3.8.1)
OLI FOV Frames	756	Frames (along-track)
WRS-2 Scene	24	Seconds (center-to-center)
	5664	Frames (center-to-center)
WRS-2 Scene Overlap	566	Frames (5% top & 5% bottom)
L0 Scene Length	6230	Frames (180km coverage, 5664+566)
	6986	Frames (adjusted for FOV, 6230+756)
	7001	15 frame cushion added (6986 + 15)
	29.67	Seconds for 7001 frames
Minimum OLI Overlap ($Start_i$-to-$Stop_{i-1}$)	1322	Frames (756 FOV + 566 overlap)

Table 4-1. OLI Level 0 Standard Length Properties

Similarly, the standard length of a full Level-0 TIRS scene is given by the following:

TIRS Sample Rate	14.286	Milliseconds / frame
TIRS FOV SCA Staggering	5.0	Degrees (along-track) The angle required for the leading and trailing imaging bands of the SCAs to cover a point on the ground relative to the center of the focal plane. See Scene Sizing Background (4.1.3.8.1)
TIRS FOV Frames	800	Frames (along-track, includes science rows)
WRS-2 Scene	24	Seconds (center-to-center)
	1680	Frames (center-to-center)
WRS-2 Scene Overlap	168	Frames (5% top & 5% bottom)
L0 Scene Length	1848	Frames (180km coverage, 1680+168)
	2648	Frames (adjusted for FOV, 1848+800)
	2760	Frames (adjusted alignment tol, 2648+112)
	2801	41 frame cushion added (2760 + 41)
	40.01	Seconds for 2621 frames
Minimum TIRS Overlap ($Start_i$-to-$Stop_{i-1}$)	1080	Frames (800 FOV + 168 overlap + 112 align)

Table 4-2. TIRS Level 0 Standard Length Properties

Note: If the instrument start and stop times between OLI and TIRS are not properly synchronized, there could be multiple partial scenes at the beginning and/or ending of the interval. Based on the start time values above, TIRS collects should start/end roughly 5 seconds before/after OLI [(40.01 – 29.67)/2 =5.17]

4.1.3.5.2 Partial Scenes

A Full WRS-2 scene product with approximately ten percent overlap is defined as 180 km in along-track direction. Additionally, the Field-of-View (FOV) offset for each instrument and the boresight misalignment is included in the minimum number of frames to ensure coverage. For OLI, this is 7001 frames, and for TIRS, this is 2801 frames. Partial WRS-2 scenes are defined as anything less than a Full WRS-2 scene. For partial scenes, the scene center is computed from the image frame closest to the nominal WRS-2 scene center. In other words, for partial scenes with more than half of a scene in length, the computed scene center is the "actual" WRS-2 scene center. For partial scenes with less than half of a scene in length, the computed scene center is the point within the imagery that is closest to the WRS-2 scene center. For short partials that are at the start of an interval, this would be at the center point of the first line, and for partials that are at the end of an interval, this would be the center point of the last line.

In addition, L8/9 have two instruments that are commanded on/off separately, so there may be times when one sensor is collecting data over a WRS-2 coverage area, while the other instrument is not. One likely scenario is that the instrument start and stop times between OLI and TIRS are not properly synchronized, e.g., the TIRS start time is more than 5 seconds before/after OLI. This could lead to "incidental" partials. Incidental partials are considered partials because the data from one of the instruments is a partial or does not exist, while data from the other instrument is full or partial over the same WRS-2 path/row. The table below defines when "incidental" partials would occur:

OLI	TIRS	Scene
Full	Full	Full
Full	Partial	Partial (incidental)
Full	None	Partial (incidental)
Partial	Full	Partial (incidental)
Partial	Partial	Partial
Partial	None	Partial (incidental)
None	Full	Partial (incidental)
None	Partial	Partial (incidental)

Table 4-3. Incidental Scene Definition

Scenes are classified as partials (incidental) even though one of the instruments may have the full WRS-2 coverage, with overlap. Currently, products are not made from incidental partials, as a combined scene collect is considered a partial. In the future, a combined scene collect with a full scene for one instrument and partial for the other may be separated such that a product can be made from the full scene.

Partials from an OLI-Only or TIRS-Only collect are not considered incidental partials.

4.1.3.5.3 Minimum Scene Overlap

WRS-2 scenes are defined to be at least 180 km long, which means approximately 10 percent overlap from scene to scene. Due to the pushbroom FOV nature of the L8/9 instruments, the start of a scene in Level-0 format needs to begin much earlier and end much later than past satellites to achieve 10 percent overlap in all bands on all Sensor Chip Assemblies (SCA). As illustrated in the following diagram, the minimum overlap requires 5 percent from each scene and ½ the FOV SCA staggering from each scene (see Figure 4-7). In other words, the minimum number of frames for overlap is 10 percent of the center-to-center frame requirement + the total FOV SCA staggering requirement. In addition, to assure minimum overlap of scenes from different orbits, the alignment counts are factored into the minimum overlap for TIRS.

For OLI-1, the center-to-center requirement is 5664 frames, making the 10 percent minimum overlap requirement 566 frames, and the FOV SCA staggering is 756 frames. Therefore, the minimum start-to-stop overlap is 566+756 = 1322 frames. The OLI-2 parameters are expected to be the same, but the FOV SCA staggering could differ by a few frames based upon the actual focal plane layout.

For TIRS-1, the center-to-center requirement is 1680 frames, making the 10 percent overlap requirement 168, and the FOV SCA staggering is 800 (including the allowance for science row deselect). In addition, the OLI-to-TIRS alignment adjustment is 112 frames. This makes the minimum start-to-stop overlap 168+800+112 = 1080 frames. Although TIRS-2 is expected to be the same, it could exhibit small differences in FOV SCA stagger and OLI-to-TIRS alignment.

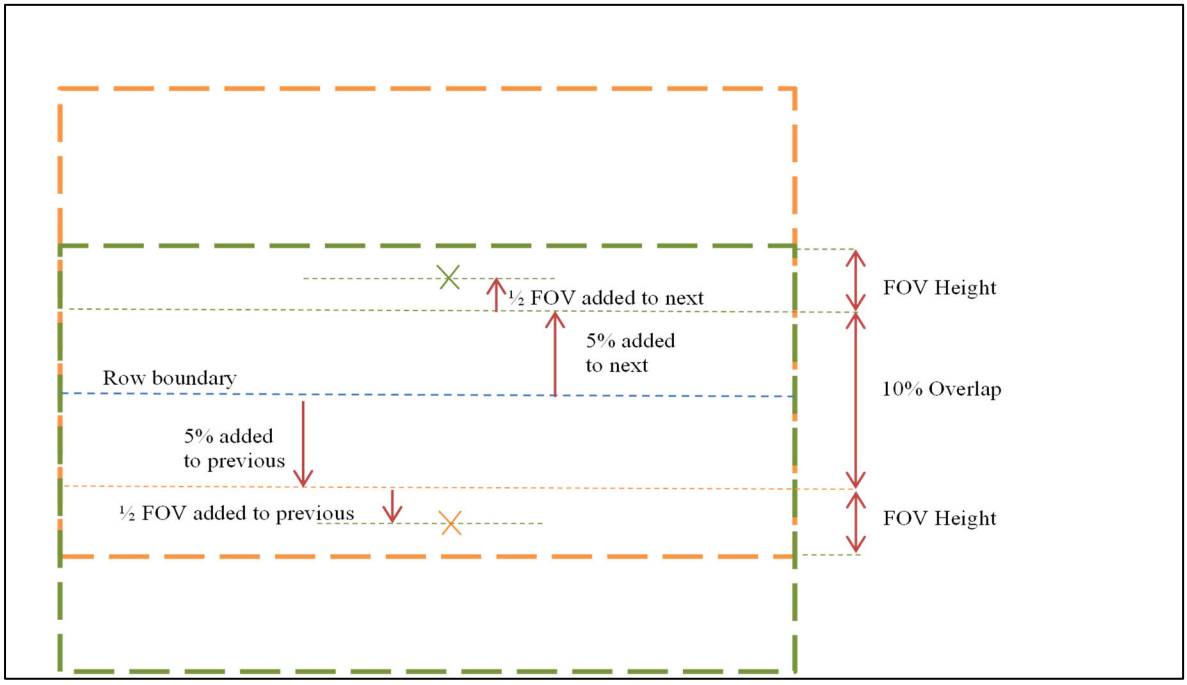

Figure 4-7. Minimum Overlap

4.1.3.5.4 Boresight Center

When the scene centers are determined from the LOS model, a boresight center is approximated from LOS projections of nearby detectors.

For OLI, the panchromatic band is the innermost band in the FOV, and SCA 7 & 8 are the closest SCAs to the center. Therefore, the left-most detector on SCA 8 and the rightmost detector on SCA 7 are used to average the latitude and longitude values obtained from the LOS projection. In Figure 4-8, the red dot marks the estimated boresight center using the projection of the two black detectors.

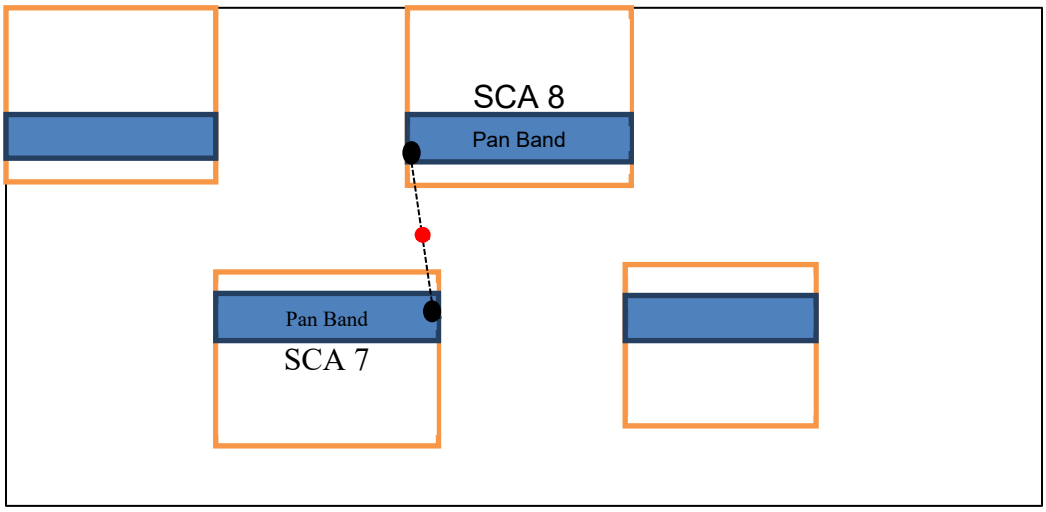

Figure 4-8. OLI Boresight Center

For TIRS, since the focal plane is made up of an odd number of SCAs, the boresight is first estimated by averaging two LOS latitude/longitude values from the outside two SCAs, then averaging the result with the center detector from the center SCA. In Figure 4-9, the blue dot represents the average of the outside two detectors, and the red dot shows the average with the center SCA's detector, yielding the boresight center estimate.

Note: The boresight estimate will vary with the actual rows selected for each SCA and will vary with latitude. Near-polar collects use an alternate method defined below.

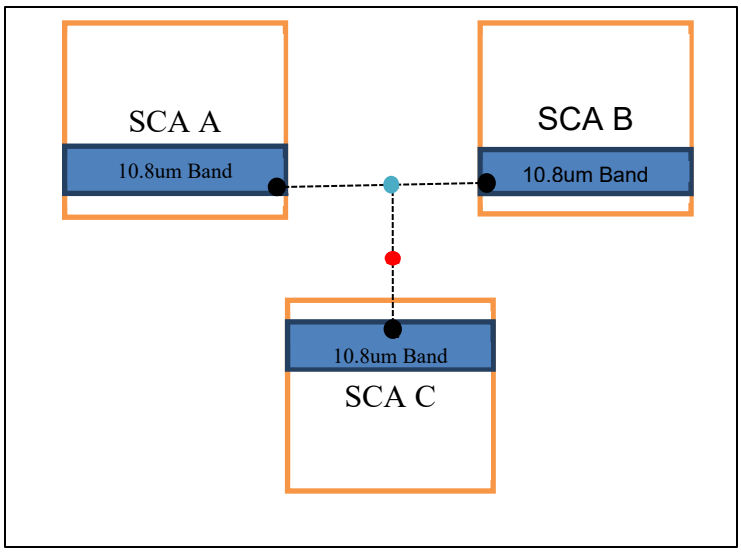

Figure 4-9. TIRS Boresight Center

4.1.3.6 Procedure

The primary tasks performed by the scene-framing algorithm include the following:

1. Load and preprocess the ancillary ephemeris and attitude data, and determine the image interval time span.
    a. The spacecraft ephemeris and attitude data from the interval ancillary data stream is quality checked and prepared for subsequent use by the ancillary data-preprocessing algorithm. The Ancillary Data Preprocessing Algorithm (see 4.1.4) describes this algorithm.
    b. Load the interval image time codes for both instruments, if present, and determine the imaging interval start and stop times.
    c. Verify that the preprocessed ancillary data completely covers the imaging interval and there is at least 8 seconds of ancillary data before and after the image data. Note: 8 seconds is expected operationally, but 4 seconds may be sufficient for processing.
2. Compute/identify orbital WRS-2 path/row (also known as nadir path/row) coverage within the imaging interval.
    a. Determine the ancillary time closest to the beginning of imaging, but not after. The ancillary time should be the latest of ACS, Ephemeris, and Inertial Measurement Unit (IMU) times, and the imaging time should be the earliest of OLI and TIRS image start times.
    b. Determine the ancillary time closest to the end of imaging, but not before. The ancillary time should be the earliest of ACS,

Ephemeris, and IMU times, and the imaging time should be the latest of OLI and TIRS image end times.

c. Use the heritage nadir scene framing algorithm Determine Nadir WRS-2 Path/Row Sub-Algorithm below and the preprocessed ancillary data to compute the starting and ending fractional WRS-2 scene path/row values. The rounded values define the orbital WRS-2 path/row span of the interval. These orbital path/rows are used for scene and interval ids even for off-nadir imaging, and for determining how the scene center times are computed.

d. Loop through the identified rows and use the heritage nadir scene framing algorithm Determine Nadir WRS-2 Path/Row Sub-Algorithm below and the preprocessed ancillary data to find the times where the fractional WRS-2 scene row values are whole numbers, i.e., where frow ≅ int(frow). These times define the initial nadir scene center times for each row.

e. Note: Because the above calculations are done based solely on the ancillary data, and ancillary data are captured before and after the imagery, the first and last scene center times calculated might fall outside of the imagery collected. These center times are adjusted below.

3. Adjust scene center times within the rows found. The initial nadir scene center times are adjusted in the following ways:

   a. For non-polar region rows (orbital WRS-2 rows from 5 through 115 and from 129 through 239)[3]; the scene center times are adjusted based on the OLI boresight[4] location.

      i. Define the OLI boresight line-of-sight vector as: $[\ 0\ 0\ 1\ ]^T$

      ii. Use the preprocessed ancillary data to interpolate the ECEF attitude quaternion at the scene center time, as described below in the Interpolate Attitude Quaternion Sub-Algorithm. Note that the scene center ECEF ephemeris will already have been computed by the nadir scene-framing algorithm.

      iii. Project the OLI boresight, to the WGS84 ellipsoid surface (i.e., using height = 0), using the algorithm described in the Forward Model, Get LOS Sub-Algorithm section of the OLI Line-of-Sight Projection/Grid Generation Algorithm Description Document (see 4.2.1).

      Note:

      - Using the nadir scene center time, found above, allows us to bypass step a).1. Find Time,

[3] Note that these rows were selected as the boundary of the polar region because a 15-degree roll at this latitude corresponds to approximately a one row offset from nadir.
[4] Use the TIRS boresight if OLI isn't present.

- Defining the OLI boresight LOS in step i. above takes the place of a).2. Find LOS,
- The quaternion interpolation of step ii. above replaces a).3. Find Attitude,
- Steps a).4. through a).7. of the Get LOS sub-algorithm can then be used to compute the geodetic latitude / longitude of the boresight intersection point.

iv. Compute the (fractional) WRS-2 path/row corresponding to the boresight latitude/longitude using the Convert Geodetic Latitude/Longitude to WRS-2 Path/Row Sub-Algorithm described below.

v. Round the row to the nearest integer. This is the adjusted row to determine a new scene center time from.

vi. Use the Search for Scene Center Time Sub-Algorithm described below to adjust the scene center time until the boresight intersection point matches the scene center adjusted row.

vii. Declare the scene center time to be the new scene center.

b. For polar rows (orbital WRS-2 rows122 and 246); the zero-crossing time of the z-component of the velocity vector from the ECEF ephemeris is the new scene center time. Note that this method will probably be close to, if not the same as, the initial nadir scene center time, but is the preferred implementation for accuracy and historical purposes.

c. For polar region rows (six rows adjacent to either side of the polar rows: orbital WRS-2 rows 1-4, 116-121, 123-128, 240-245, and 247-248); the initial nadir scene center times will not be adjusted due to the large amount of overlap of path/rows at the pole. The new scene center time will be set to the nadir scene center time found above.[5]

4. Compute the scene extents

a. Determine the scene center frames for each instrument. If the first scene's center is before the start of imagery, a negative frame number is estimated. If the last scene's center is after the end of imagery, a frame number larger than the interval length is used. These are temporary values used in the next step.

b. Compute scene start and stop frames:

i. start $frame_i$ = max(0, $center\_frame_i$ – (NOMINAL_SIZE/2)).

[5] Note that the number of adjacent rows may have to be adjusted if the polar region boundary scene sizes do not frame to the proper sizes.

ii. stop $frame_i$ =

min(total_frames, center_$frame_i$ + (NOMINAL_SIZE/2)).

c. Adjust the first and last scene center frames to be within the actual imagery, if needed (i.e., if first center < 0, make it 0 and if the last center > interval last frame, make it the last frame).

d. Check for adequate overlap between scenes, and adjust if needed:

overlap = scene stop $frame_i$ - scene start $frame_{i+1}$.

if(overlap < minimum overlap)
delta = (minimum overlap – overlap)
start diff = (delta) / 2;
if(start diff >= scene start $frame_{i+1}$)
start diff = scene start $frame_{i+1}$;
scene start $frame_{i+1}$ -= start diff;
stop diff = (delta – start diff);
scene stop $frame_i$ += stop diff;
if (scene stop $frame_i$ > total_frames)
scene stop $frame_i$ = total_frames;

e. See if the first and last partials are completely within the overlap regions of adjacent scenes. If so, remove them.

f. Set the scene start, stop, and center times to the corresponding frame times.

g. Determine which scenes are partial or full.

i. length = stop $frame_i$ – start $frame_i$;

ii. if (length < NOMINAL_SIZE)
scene is PARTIAL
else
scene is FULL

iii. Note that for combined (OLI + TIRS) collects, both lengths must be greater than or equal to the corresponding nominal sizes for the scene to be FULL.

5. Check the complete list of scene center times to ensure that no two adjacent scene centers are more than 48 seconds apart (two times the normal scene center-to-center interval). If any two consecutive scene centers exceed this limit, error out (the polar region will need to be enlarged). **This should never happen.**

6. Compute the corresponding *target* WRS-2 path/row coordinates and lat/long for each adjusted scene center time in the interval (new scene center times from step 3-4 above).

a. Define the OLI boresight line-of-sight vector as: $[\ 0\ 0\ 1\ ]^T$

b. Use the preprocessed ancillary data to interpolate the ECEF attitude quaternion at the adjusted scene center time, as described in the Interpolate Attitude Quaternion Sub-Algorithm below.

c. Project the OLI boresight, to the WGS84 ellipsoid surface (i.e., using height = 0), using the algorithm described in the Forward Model Get LOS Sub-Algorithm section of the OLI Line-of-Sight Projection/Grid Generation Algorithm Description Document (see 4.2.1).

   Note:

   - Using the adjusted scene center time, found above, allows us to bypass step a).1. Find Time,
   - Defining the OLI boresight LOS in step i. above takes the place of a).2. Find LOS,
   - The quaternion interpolation of step ii above replaces a).3. Find Attitude,
   - Steps a).4 through a).7 of the Get LOS sub-algorithm can then be used to compute the geodetic latitude and longitude of the boresight intersection point.

d. Compute the (fractional) WRS-2 path/row corresponding to the boresight latitude/longitude using the Convert Geodetic Latitude/Longitude to WRS-2 Path/Row Sub-Algorithm described below. Round the values to the nearest integers. This is the target path/row.

e. Determine if the scene center position is off the WRS-2 grid. For collections near the poles, it is possible to look off-nadir toward the pole, into an area not defined by the WRS-2 grid. If the geodetic latitude just calculated is above 82.61 degrees, this is considered off the WRS-2 grid and a special naming convention is used. To allow unique target row assignments, the North Pole area is assigned a row of 88n, and the South Pole area is assigned a row of 99n, where n is a sequential number. Up to seven scenes can be covered in these areas; therefore, the scenes are assigned target row numbers 880 to 886, or 990 to 996 in the interval.

4.1.3.7 Corner Coordinate Framing

The corner points represent the WRS-2 extent of a scene on the ground in north-up latitude and longitude coordinates. Using the scene's starting and ending frames, found above, a Line-of-Sight is calculated at the first and last pixel in those frames (use the Forward Model Get LOS Sub-Algorithm section of the OLI Line-of-Sight Projection/Grid Generation Algorithm Description Document (see 4.2.2) and TIRS Line-of-Sight Projection/Grid Generation (see 4.3.2). Due to the layout of the bands and SCAs on the focal plane, there are along-track offsets between bands within each SCA, along-track offsets between even and odd

SCAs, and a reversal of the band ordering in adjacent SCAs. To create more uniform image coverage, the leading and trailing imagery associated with these offsets is “trimmed” based on an active area bounds.

To account for band offsets, the frames and pixels from the outermost bands should be used in the corner calculations. For the OLI corner calculations, the Cirrus band is used, and similarly, the 10.8 µm band is used for TIRS. Using the outermost bands ensures that every band is bounded by the corner coordinates. To account for the SCA offsets, a minbox representing a rectangular active image frame is defined that excludes the excess trailing imagery from even SCAs, and the excess leading imagery from odd SCAs.

4.1.3.7.1 OLI Active Image Area

The active image area (minbox) for OLI is computed by constructing 8 critical SCA corner points from the Cirrus band, labeled C1 through C8 in Figure 4-10. Points C1 and C2 define the top edge of the active area, C3 and C4 the right edge, C5 and C6 the bottom edge, and C7 and C8 the left edge. Note that points C1 and C8 are the same (the upper-left corner of SCA01), as are points C4 and C5 (the lower-right corner of SCA14). Use the forward model to project these eight line/sample locations to object space, computing the latitude/longitude coordinates for each point. The average elevation over the WRS-2 path/row is used as a rough adjustment from the WGS84 ellipsoid in the elevation parameter of the forward model for the eight line/sample to latitude/longitude calculations. Use the WRS-2 Scene average elevation look-up file to determine the average elevation for the path/row being processed.

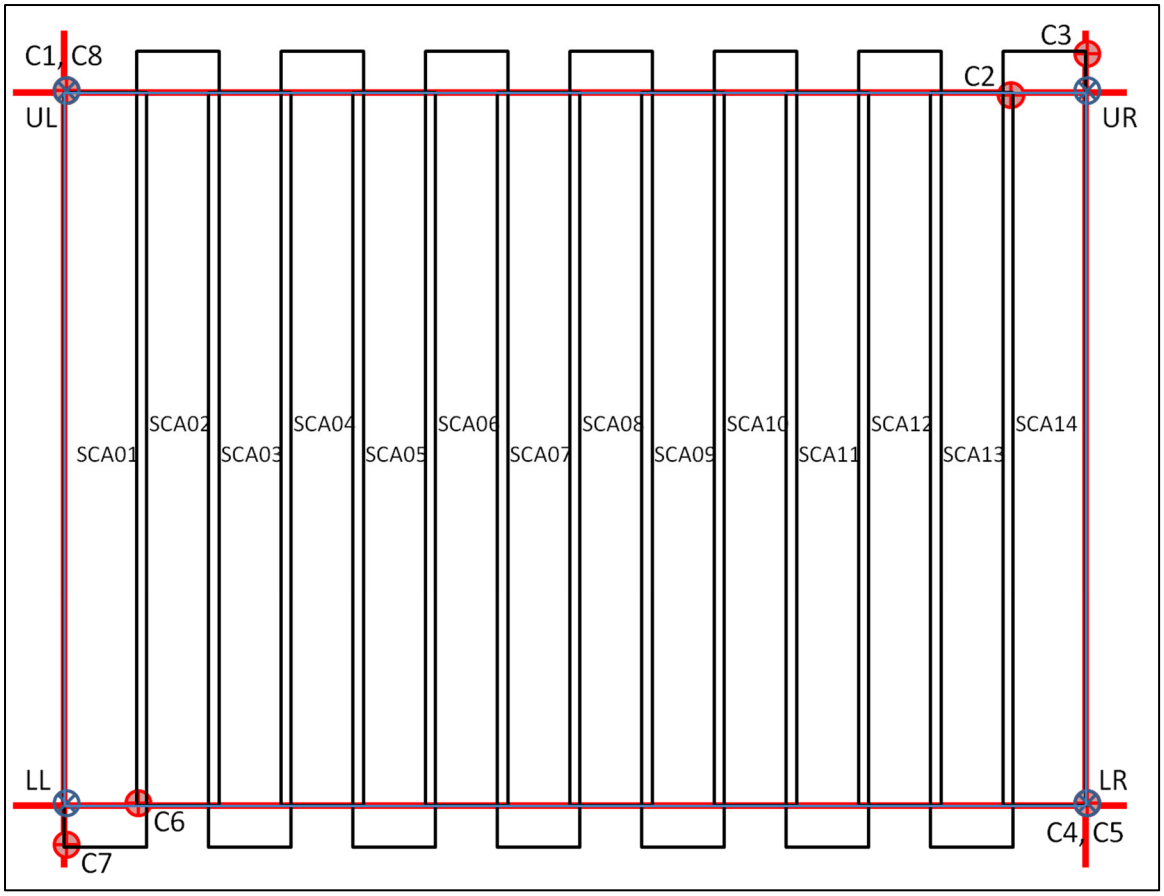

Figure 4-10. Active OLI Image Area

The remainder of the calculations and determination of the minbox framing of the active OLI image area is described in the Calculating the Active Image Area section of the OLI Line-of-Sight Projection/Grid Generation Algorithm (see 4.2.2). The results of these calculations should be the latitude and longitude of the four bounding corner points represented by the blue points in Figure 4-10.

4.1.3.7.2 TIRS Active Image Area

The active image area (minbox) for TIRS is computed by constructing 8 critical SCA corner points from the 10.8 µm band, labeled C1 through C8 in Figure 4-11. This figure depicts the current understanding of the TIRS field of view orientation with respect to object space, but the algorithm described here will work as long as the SCAs are numbered sequentially across the field of view, in either direction. Points C1 and C2 define the top edge of the active area, C3 and C4 the right edge, C5 and C6 the bottom edge, and C7 and C8 the left edge. Note that points C4 and C5 are the same (the lower- right corner of SCA01), as are points C6 and C7 (the lower-left corner of SCA03). Use the forward model to project these eight line/sample locations to object space, computing the latitude/longitude coordinates for each point. The average elevation over the WRS-2 path/row is used as a rough adjustment from the WGS84 ellipsoid in the elevation parameter of the forward model for the eight line/sample to

latitude/longitude calculations. Use the WRS-2 Scene average elevation look-up file to determine the average elevation for the path/row being processed.

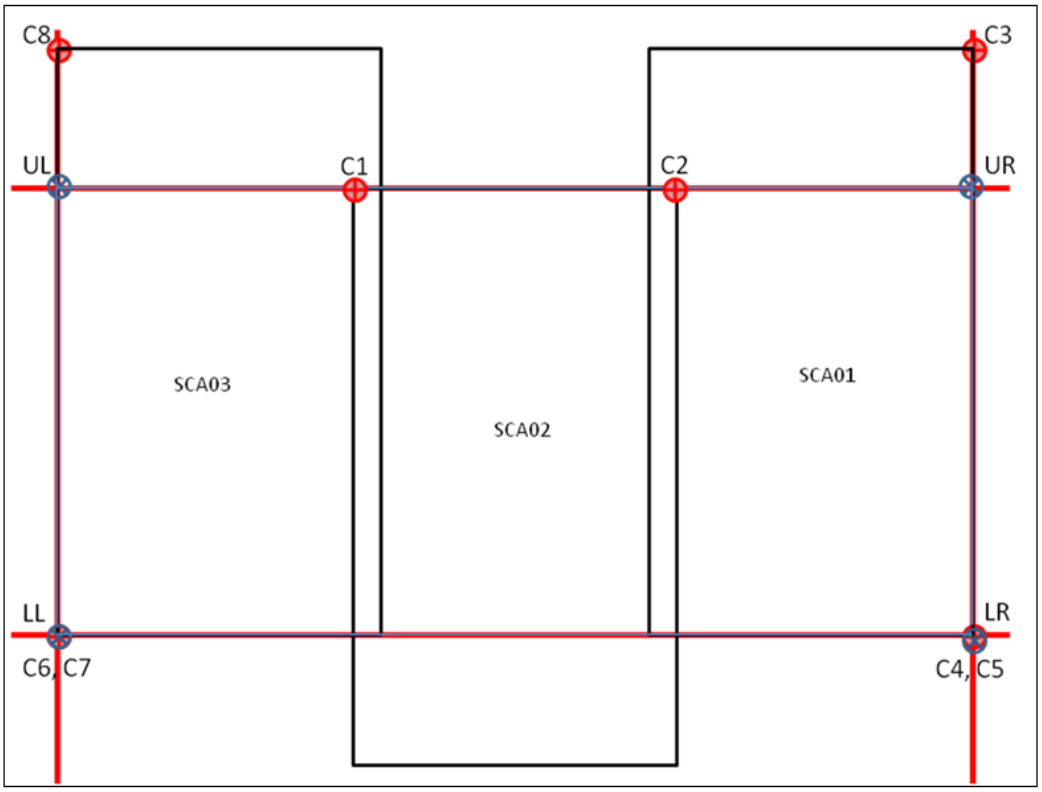

Figure 4-11. Active TIRS Image Area

The corner point assignments are made by examining the SCA across-track and along-track Legendre coefficients to determine: 1) whether SCA01 is on the right (-Y) or left (+Y) side of the scene; 2) whether even or odd SCAs lead; and 3) whether the sample number increases in the –Y or +Y direction. If the across-track Legendre constant term (coef_y0) for SCA01 is positive, then it is the left-most SCA and SCA03 is the right-most. If the along-track Legendre constant term (coef_x0) for SCA01 is greater than that for SCA02, then the odd SCAs lead. If the across-track Legendre linear term (coef_y1) for SCA01 is negative, then the sample number increases in the –Y direction.

Having determined the orientation of the SCAs, assign the top edge to the left-most leading SCA Upper-Left (UL) corner and the right-most leading SCA Upper-Right (UR) corner, the right edge to the right-most SCA UR and Lower-Right (LR) corners, the bottom edge to the right-most trailing SCA LR corner and left-most trailing SCA lower-left (LL) corner, and the left edge to the left-most SCA LL and UL corners. As shown in the figure, for the TIRS: C1 = SCA02 (left-most leading SCA) UL, C2 = SCA02 (right-most leading SCA) UR, C3 = SCA01 (right-most

SCA) UR, C4 = SCA01 (right-most SCA) LR, C5 = SCA01 (right-most trailing SCA) LR, C6 = SCA03 (left-most trailing SCA) LL, C7 = SCA03 (left-most SCA) LL, and C8 = SCA03 (left-most SCA) UL. Note that these assignments are based on the current TIRS SCA ordering of SCA-B = SCA01, SCA-C = SCA02, and SCA-A = SCA03, and could change if the SCA numbering system is revised. If this were to happen, the change would be reflected in the Legendre coefficients, so the logic described here would automatically compensate.

The Calculating the Active Image Area section of the TIRS Line-of-Sight Projection/Grid Generation Algorithm (see 4.3.2) describes the remainder of the calculations and determination of the minbox framing of the active TIRS Image area. The results of these calculations should be the latitude and longitude of the four bounding corner points represented by the blue points in Figure 4-11.

As depicted in Figure 4-12, the lower corner coordinates correspond to the leading edge (last line) of a scene, and upper coordinates correspond to the trailing edge (first line) of a scene from the outermost bands on the SCAs. Leading/Trailing edges are based on which SCA/Band/Detectors are first/last in relation to the direction of flight (ascending or descending) relative to the ground.

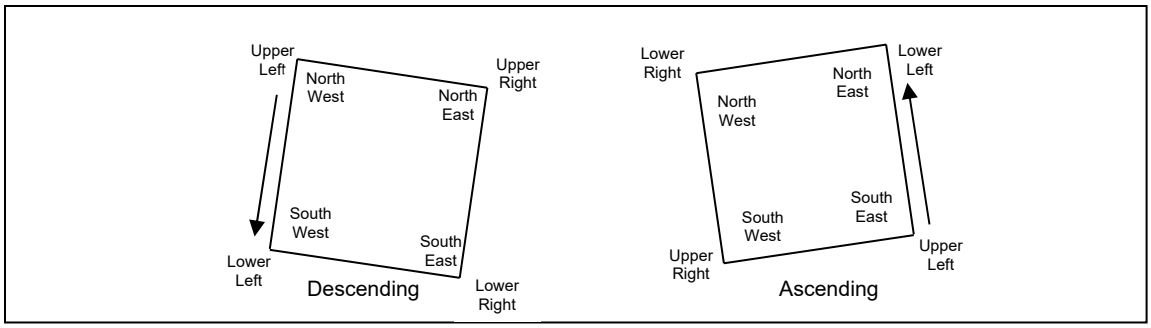

Figure 4-12. Leading/Trailing Scene Edge

4.1.3.7.3 Determine Nadir WRS-2 Path/Row Sub-Algorithm

The ephemeris data are used to define the nadir WRS-2 path and row. The following routine is called to determine the nadir-pointing position of the satellite for Landsat Scene IDs and to determine scene center times for polar region rows. This is also known as the “heritage nadir scene framing algorithm.”

Inputs:

- *ecef_pos, ecef_vel* (Ephemeris State Vector in Earth-Centered, Earth-Fixed coordinates).
- CPF WRS-2 Constants:
  - $Long\_Path_1\_Row_{60}$ (longitude of Path 1 at Row 60 = -64.6 deg).
  - *WRS_Cycle_Days* (number of days per WRS cycle = 16 days)
  - *WRS_Cycle_Orbits* (number of orbits per WRS cycle = 233 orbits)
  - *Scenes_Per_Orbit* (number of scenes or rows in each orbit = 248 rows)

- *Descending_Node_Row* (row number of equator when descending = 60)
- *Omega_E* (WGS-84 Earth's inertial rotation rate, rad/sec)

Outputs:

- Fractional WRS-2 Orbital Path & Row.

Procedure:

1. Convert the CPF path 1 row 60 longitude to radians.

$$Long\_Path_1\_Row_{60} = \frac{Long\_Path_1\_Row_{60} * \pi}{180}$$

2. Compute the Earth angular rate (solar to account for orbital precession).

$$Earth\_Spin\_Rate = \frac{2*\pi}{24*3600}$$

3. Compute the spacecraft angular rate.

$$SC\_Ang\_Rate = \frac{2*\pi*WRS\_Cycle\_Orbits}{WRS\_Cycle\_Days*24*3600}$$

4. Normalize the incoming position and velocity vectors.

$$mag = \sqrt{ecef\_pos_x{}^2 + ecef\_pos_y{}^2 + ecef\_pos_z{}^2}$$

$$sc\_pos._x = \frac{ecef\_pos_x}{mag}$$

$$sc\_pos_y = \frac{ecef\_pos_y}{mag}$$

$$sc\_pos_z = \frac{ecef\_pos_z}{mag}$$

Adjust the velocity vector by Earth's inertial rotation rate. Note: If the ephemeris data have already been preprocessed, the ADP output ECEF ephemeris can be used, and this velocity adjustment is not needed.

$$new\_vel._x = ecef\_vel_x - omega\_e * ecef\_pos_y$$

$$new\_vel._y = ecef\_vel_y + omega\_e * ecef\_pos_x$$

$$new\_vel._z = ecef\_vel_z$$

and normalize.

$$mag = \sqrt{new\_vel_x{}^2 + new\_vel_y{}^2 + new\_vel_z{}^2}$$

$$sc\_vel_x = \frac{new\_vel_x}{mag}$$

$$sc\_vel_y = \frac{new\_vel_y}{mag}$$

$$sc\_vel_z = \frac{new\_vel_z}{mag}$$

5. Compute the spacecraft angular momentum $R \times Y$.

$$ang\_mo = \left| sc\_pos \times sc\_vel \right|$$

6. Compute the vector to the descending node.

$$dnode_x = \frac{ang\_mo_y}{\sqrt{ang\_mo_x{}^2 + ang\_mo_y{}^2}}$$

$$dnode_y = \frac{-ang\_mo_x}{\sqrt{ang\_mo_x{}^2 + ang\_mo_y{}^2}}$$

$$dnode_z = 0$$

and normalize.

$$mag = \sqrt{dnode_x{}^2 + dnode_y{}^2 + dnode_z{}^2}$$

$$dnode_x = \frac{dnode_x}{mag}$$

$$dnode_y = \frac{dnode_y}{mag}$$

$$dnode_z = \frac{dnode_z}{mag}$$

7. Compute the central travel angle from the descending node and the spacecraft position vector.

$$temp = dnode \times sc\_pos$$
$$mag = |temp|$$
$$s = \text{sign}(temp \bullet ang\_mo)$$
$$central\_angle = \text{atan}\,2(mag * s, (dnode \bullet sc\_pos))$$

8. Compute the row number from the central angle.

$$frow = Descending\_Node\_Row + \frac{central\_angle}{2 * \pi} * Scenes\_Per\_Orbit$$
$$frow < 0.5 \Rightarrow frow = frow + Scenes\_Per\_Orbit$$
$$frow > (Scenes\_Per\_Orbit + 0.5) \Rightarrow frow = frow - Scenes\_Per\_Orbit$$

9. Compute the longitude of the instantaneous descending node.

$$inst\_dnode\_long = \text{atan}\,2(-ang\_mo_x, ang\_mo_y)$$
$$long\_origin = inst\_dnode\_long + \frac{central\_angle * Earth\_Spin\_Rate}{SC\_Ang\_Rate}$$

10. Compute the path number from the longitude of row 60.

$$fpath = \frac{Long\_Path1\_Row60 - long\_origin}{2 * \pi} * WRS\_Cycle\_Days + 1$$
$$(central\_angle < \frac{(0.5 - Descending\_Node\_Row) * 2 * \pi}{Scenes\_Per\_Orbit}) \Rightarrow fpath = fpath - 16$$
$$(fpath < 0.5) \Rightarrow fpath = fpath + WRS\_Cycle\_Orbits$$

Note: The (0.5 – Descending_Node_Row) is the distance in WRS rows from the start of the path (row 0.5) to the descending node (row 60).

11. Make sure the row number is in range.

```
while ( frow < 0.5 )
    fpath = fpath – 16;
    frow = frow + Scenes_Per_Orbit;

while ( frow > Scenes_Per_Orbit + 0.5 )
    fpath = fpath + 16;
    frow = frow – Scenes_Per_Orbit;
```

12. Make sure the path number is in range.

```
while ( fpath < 1 )
```

fpath = fpath + WRS_Cycle_Orbits;

while (fpath > WRS_Cycle_Orbits)
fpath = fpath – WRS_Cycle_Orbits;
frow = frow – Scenes_Per_Orbit;

4.1.3.7.4 Interpolate Attitude Quaternion Sub-Algorithm

Given a sequence of time-stamped quaternions, ($\mathbf{q}_i$, t_i), and a time, t_0, at which the interpolated quaternion is desired:

1. Step through the quaternion time stamps to identify the latest quaternion time, t_i, which is less than or equal to the interpolation time of interest, t_0.
2. Calculate the quaternion $\Delta\mathbf{q}$ that rotates $\mathbf{q}_i$ to $\mathbf{q}_{i+1}$:
   $\Delta\mathbf{q} = \mathbf{q}_{i+1}\ \mathbf{q'}_i$
   where: $\mathbf{q'}_i$ is the conjugate of quaternion $\mathbf{q}_i$. See the quaternion conjugation and quaternion multiplication sub-algorithms below.
3. If the sign of the fourth element of $\Delta\mathbf{q}$, Δq_4, is negative, change the sign of the entire quaternion.
4. Decompose the $\Delta\mathbf{q}$ quaternion into its angle (θ) and axis of rotation ($\mathbf{x}$) form:
   $\sin(\theta/2) = \text{sqrt}(\Delta q_1 * \Delta q_1 + \Delta q_2 * \Delta q_2 + \Delta q_3 * \Delta q_3)$
   $\cos(\theta/2) = \Delta q_4$
   $\theta = 2 * \text{atan}(\sin(\theta/2) / \cos(\theta/2))$
   $\mathbf{x} = [\ \Delta q_1/\sin(\theta/2)\ \ \Delta q_2/\sin(\theta/2)\ \ \Delta q_3/\sin(\theta/2)\ ]^T$
   noting that if $\sin(\theta/2) = 0$ then $\mathbf{x} = \mathbf{0}$.
5. Linearly interpolate the angle θ_0 at time t_0:
   $\theta_0 = \theta * (t_0 - t_i) / (t_{i+1} - t_i)$
6. Construct the quaternion corresponding to the new rotation angle θ_0:
   $\Delta\mathbf{q_0} = [\ \sin(\theta_0/2)\ \mathbf{x}^T\ \ \cos(\theta_0/2)\ ]$
7. Apply the new delta quaternion to $\mathbf{q}_i$ to compute $\mathbf{q_0}$, the quaternion at time t_0:
   $\mathbf{q_0} = \Delta\mathbf{q_0}\ \mathbf{q}_i$

Quaternion Conjugation Sub-Algorithm
The conjugate **q'**, of a quaternion **q**, is computed by inverting the sign on the first three elements of **q**:

$q' = [\ -q_1\ \ -q_2\ \ -q_3\ \ q_4\ ]$

Quaternion Multiplication Sub-Algorithm
The product **c**, of quaternions **a** and **b** is given by:

$c_1 = a_4 b_1 + a_3 b_2 - a_2 b_3 + a_1 b_4$
$c_2 = -a_3 b_1 + a_4 b_2 + a_1 b_3 + a_2 b_4$
$c_3 = a_2 b_1 - a_1 b_2 + a_4 b_3 + a_3 b_4$
$c_4 = -a_1 b_1 - a_2 b_2 - a_3 b_3 + a_4 b_4$

Note: Quaternion multiplication does not commute. Also, note that other formulations of quaternion multiplication are possible. Any consistent formulation should work in the interpolation equations above.

4.1.3.7.5 Convert Geodetic Latitude/Longitude to WRS-2 Path/Row Sub-Algorithm

Given a boresight geodetic latitude ϕ, and longitude λ, and the corresponding spacecraft velocity Z component V_z (to determine whether the scene is ascending or descending):

1. Compute the geocentric latitude θ, from the geodetic latitude ϕ, and the WGS84 ellipsoid semi-major (a) and semi-minor (b) axes:

   θ = atan(tan(ϕ) * b/a * b/a)

2. Use the geocentric latitude and the nominal Landsat orbital inclination (i = 98.2 degrees) to compute the longitude offset to the apparent descending node:

   λ_{off} = asin(tan(θ) / tan (π - i))

   This calculation should include a test to ensure that the argument of the asin function is in the range -1 to +1, clipping the value to this range if necessary (e.g., for latitudes outside the standard WRS-2 range).

3. Calculate the central travel angles for both the descending and ascending cases:

   CTA_d = asin(-sin(θ) / sin (π - i))
   CTA_a = π - CTA_d

   As above, the range of the asin function argument should be clipped to the range -1 to +1.

4. Compute the allowable range of central travel angles on a given WRS path as:

   min CTA = (0.5 - 60.0)/248 * 2π
   max CTA = (248.5 - 60.0)/248 * 2π

5. Add or subtract 2π to the descending and ascending central travel angles to bring them within the allowable range.

   If (CTA_d < min CTA) CTA'_d = CTA_d + 2π
   Else if (CTA_d > max CTA) CTA'_d = CTA_d - 2π
   Else CTA'_d = CTA_d
   If (CTA_a < min CTA) CTA'_a = CTA_a + 2π
   Else if (CTA_a > max CTA) CTA'_a = CTA_a - 2π
   Else CTA'_a = CTA_a

6. Compute the Earth rotation angles from the adjusted central travel angles:

$\Delta\lambda_d$ = CTA'$_d$ * Earth rotation rate / Spacecraft angular rate
$\Delta\lambda_a$ = CTA'$_a$ * Earth rotation rate / Spacecraft angular rate

7. Calculate the apparent descending node longitudes for the descending and ascending cases:

   $DN\lambda_d = \lambda - \lambda_{off} + \Delta\lambda_d$
   $DN\lambda_a = \lambda + \lambda_{off} + \pi + \Delta\lambda_a$

8. Select the descending or ascending case:

   If (Vz > 0.0)
   CTA' = CTA'$_a$
   $DN\lambda = DN\lambda_a$
   Else
   CTA' = CTA'$_d$
   $DN\lambda = DN\lambda_d$

9. Compute the (fractional) WRS-2 row number from the central travel angle:

   row = row at equator (60) + CTA' / 2π * number of rows (248)

10. Compute the (fractional) WRS-2 path number:

    λ_0 = (longitude of path 1 at equator) - $DN\lambda$ + 2π
    If ($\lambda_0 > 2\pi$) $\lambda_0 = \lambda_0 - 2\pi$
    path = λ_0 * number of paths (233) / 2π + 1

4.1.3.7.6 Search for Scene Center Time Sub-Algorithm

Given a scene center time (t_0) and the corresponding WRS-2 row (row_0) and a target (integer) row (row_T):

1. Compute the nominal WRS-2 row rate:

   row rate = number of rows (248) / orbital period

2. Compute the row error:

   row error = row_0 - row_T

3. If the absolute value of the row error is less than 0.005, use the current scene center and exit the search.

4. Save the previous scene center time and row:

   $row_L = row_0$
   $t_L = t_0$

5. Adjust the scene center time:

   $t_0 = t_0$ - (row error) / (row rate)

6. Interpolate the spacecraft ephemeris and attitude at the new scene center time.

7. Project the boresight to the WGS84 ellipsoid at the new scene center time.
8. Compute the WRS-2 path/row at the boresight latitude/longitude as described above. This yields a new value of row_0.
9. Compute the WRS-2 row rate:

   row rate = $(row_0 - row_L) / (t_0 - t_L)$
10. Continue the iterations at step 2 above.

Note: This sub-algorithm is only used for non-polar scenes, so the row transition from 248 to 1 is not an issue.

4.1.3.8 Notes

Significant algorithm assumptions and notes, including those embedded in the text above, are as follows:

1. Ancillary data for the full imaging interval with 8 seconds of extra before and after the interval are available to provide the required geometric support data. A minimum of 4 seconds of extra ancillary data before and after **may** be sufficient for processing.
2. In the polar transition regions (rows 5, 115, 129, and 239), there may be larger scenes framed in situations where the spacecraft is rolled "away" from the pole (see step 2.d. of the main algorithm procedure above). In this configuration, the distance between scene centers can exceed 5664 OLI multispectral image frames. This occurs because the OLI is looking toward the equator where the rows are growing farther apart in latitude, while the spacecraft is flying at higher latitude where the rows are closer together in latitude. It thus takes more than a nominal row of flight time for the boresight to traverse one row of latitude. Other possible approaches would be to move the transition region toward the equator by five rows for intervals that are rolled toward the equator. The approach adopted here is to allow the off-nadir Level 0 Reformatted Product (L0Rp) scenes to be somewhat longer than nadir scenes.

4.1.3.8.1 Scene Sizing Background

4.1.3.8.1.1 OLI

The boresight of the OLI telescope is parallel to the spacecraft +Z axis, which means it will be nadir pointing. The 14 OLI SCAs are arranged in two rows of seven, as shown in Figure 4-13. The Field of View (FOV) of the telescope is 15 degrees in the cross-track direction and 1.7 degrees in the along-track direction.

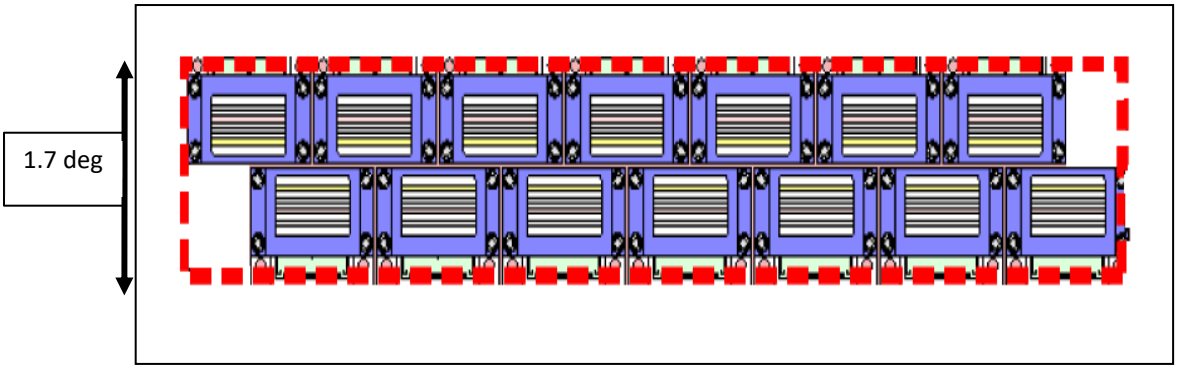

Figure 4-13. OLI SCA Layouts and FOV

The telescope boresight will traverse one scene (i.e., scene center to scene center) in about 24 seconds, given the nominal orbit rate.

WRS_Cycle_Days = 16 days
WRS_Cycle_Orbits = 233
Scenes_Per_Orbit = 248
Seconds_Day = 86400

Calculate Spacecraft Angular Rotational rate:

$$\omega = \frac{2 * \pi * WRS\_Cycle\_Orbits}{WRS\_Cycle\_Days * Seconds\_Day} = 0.0010591049 \frac{radians}{seconds}$$

Calculate time between successive WRS rows:

$$\triangle t = \frac{WRS\_Cycle\_Days\ * Second\_Day}{WRS\_Cycle\_Orbits * Scenes\_Per\_Orbit} = 23.923577\ seconds$$

The size of an OLI scene, in lines, can be calculated with respect to the boresight with the following calculations. Rounding the above number to 24 seconds, the number of OLI lines between successive WRS rows is as follows:

OLI_Frame_Rate = 236 lines / second

$$OLI\ Lines = \frac{\triangle t}{OLI\_Frame\_Rate} = 5664\ Lines$$

In addition, by definition of Landsat WRS products, WRS scenes are overlapped with the previous row by 5 percent, and the next row by 5 percent. The total number of OLI lines needed for the overlap is then 10 percent.

$$OLI\ Lines = OLI\ Lines + 0.1 * \frac{\Delta t}{OLI\_Frame\_Rate} = 6230\ Lines$$

However, the L8/9 OLI bands within each SCA are staggered with respect to the boresight.

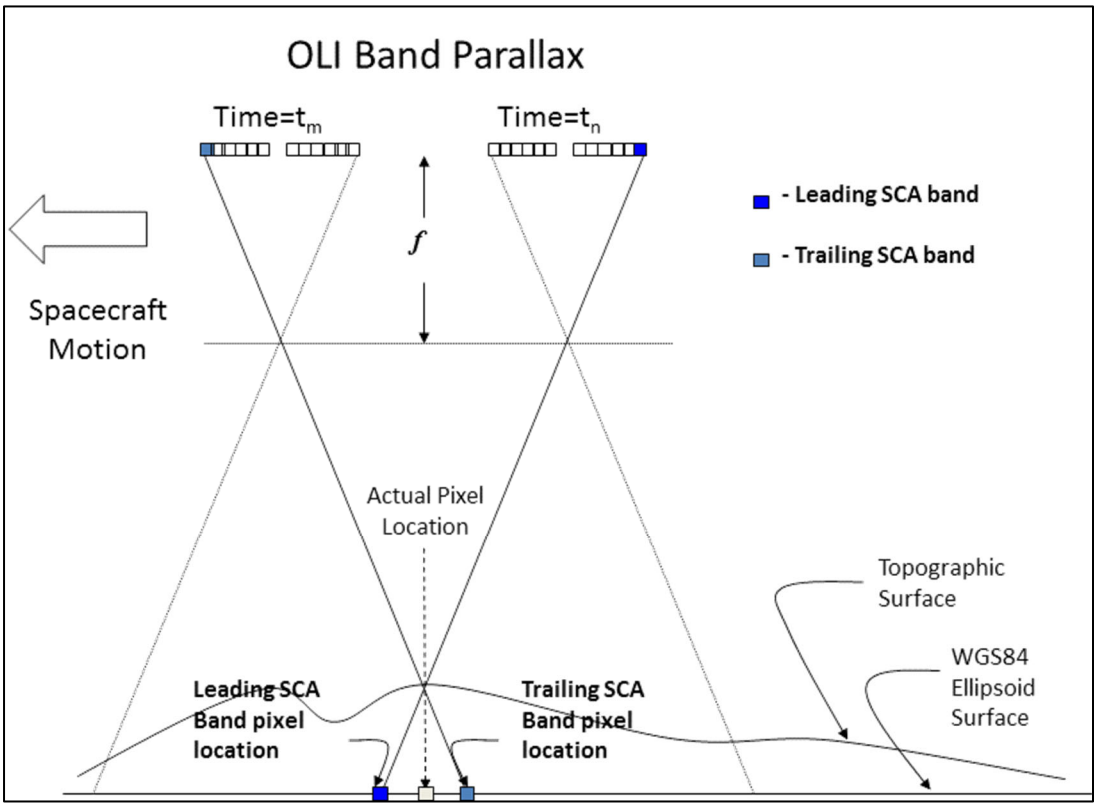

Figure 4-14. OLI Band/SCA Band Parallax

This staggering represents a time difference between when the leading set of detectors within a SCA, for a given band, image the start and end of the WRS scene, and when the trailing set of detectors within a SCA, for a given band, image the start and end of the WRS scene.

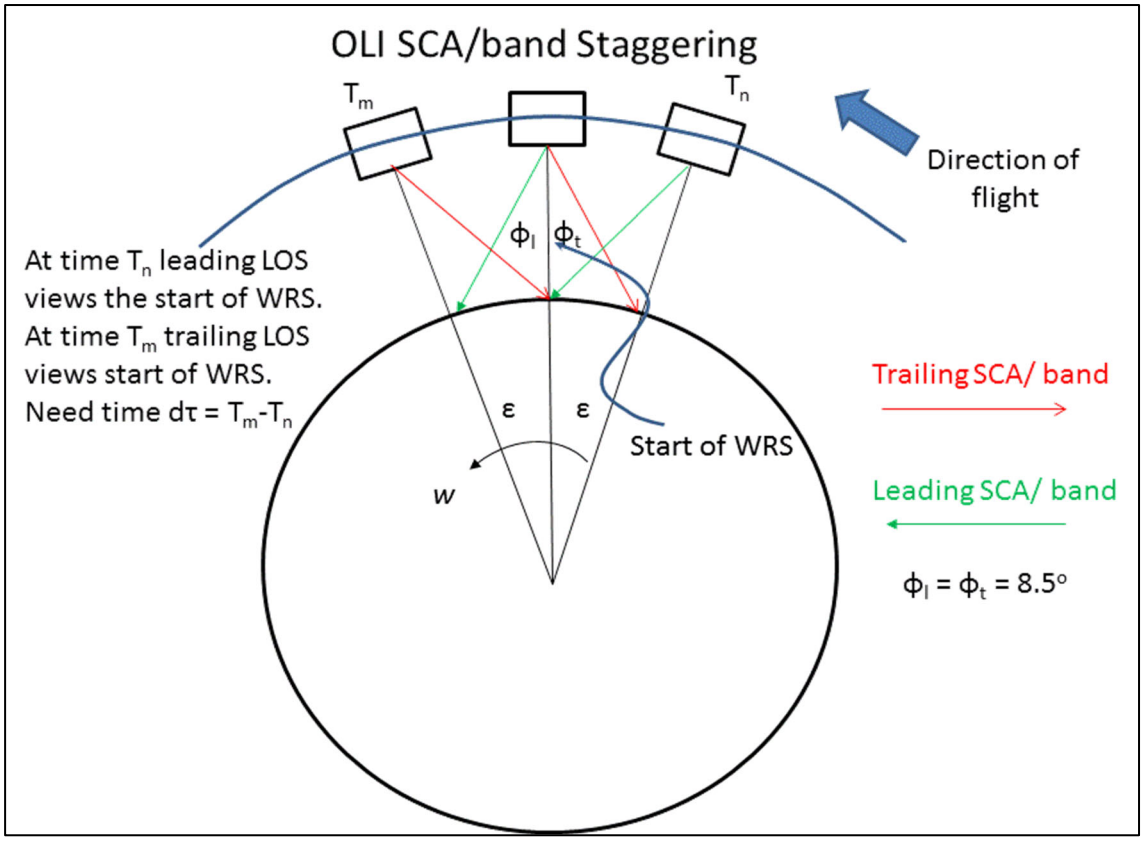

Figure 4-15. OLI SCA/Band Staggering

The extra time required for the leading and trailing imaging bands of the SCAs to cover a point on the ground relative to the center of the focal plane will vary with position in orbit and scene elevation. Extra lines of imaging will be required at the beginning and end of the interval, or about 1.5 seconds of extra data on each side. This means that imaging must start at least 1.5 seconds prior to the telescope boresight reaching the leading edge of the first scene in the desired interval and must continue for at least 1.5 seconds after passing over the trailing edge of the last scene of an interval. This will assure that all bands in all SCAs have completely imaged the scene.

Looking at the OLI Legendre LOS polynomials and determining the leading and trailing look vectors, the difference in radians/degrees is as follows:

Leading_LOS = 1.436414e-02 radians = 0.8230046 degrees
Trailing_LOS = -1.444101e-02 radians = -0.8265414 degrees

The difference between these two numbers represents a field of view of 1.65 degrees. Rounding this to 1.7 degrees (0.85 leading and 0.85 trailing), we can determine the amount of time needed to pad either the leading or trailing acquisition of the OLI WRS scene in terms of time. If the maximum satellite altitude is present at the poles and the minimum satellite altitude is present at the

equator, the minimum and maximum number of lines needed in the L0Rp can be calculated.

α = 0.85 degrees
Orbital_Radius = 7083445.719 meters
Semi_Major_Axis = 6378137.0 meters
Semi_Minor_Axis = 6356752.3142 meters
Major_Alt = Orbital_Radius – Semi_Major_Axis = 705308.72 meters
Minor_Alt = Orbital_Radius – Semi_Minor_Axis = 726693.40 meters

The ground distance covered at these two altitudes represented on the Earth are found as follows:

$$GD\_Major = Major\_Alt * \tan\ \alpha = 10464.234\ \ meters$$
$$GD\_Minor = Minor\_Alt * \tan\ \alpha = 10781.505\ \ meters$$

These ground distances represent an angular orbit change of the following:

$$\varepsilon = \text{atan}\left(\frac{GD\_Major}{Semi\_Major\_Axis}\right) = 0.00164063\ radians$$

$$\varepsilon = \text{atan}\left(\frac{GD\_Minor}{Semi\_Minor\_Axis}\right) = 0.00169607\ radians$$

Using the spacecraft angular rotational rate gives a delta time due to the staggering of the SCAs as the following:

$$d\tau = \frac{\varepsilon}{\omega} = 1.5\ and\ 1.6\ seconds$$

Using the OLI frame rate, the number of OLI lines needed for this (maximum) delta time is as follows:

$$\triangle\ Lines\ SCA\ Staggering = 1.6\ seconds * OLI\_Frame\_Rate = 378\ Lines$$

The total number of lines needed within an OLI WRS scene is then

Total Lines = (nominal size + 5% overlap + SCA staggering)

$$OLI\ Lines = OLI\ Lines + 2 * \triangle\ Lines\ SCA\ Staggering = 6986\ Lines$$

For convenience and ease of use, the final number of OLI lines will be rounded to 7001 lines. An odd number is used so that a center line is found and the same number of before and after lines (3500) are used to define the entire L0R scene.

In terms of time, a single scene will take about 29.6 seconds:

1.6 sec (378 lines for leading edge SCA coverage)
\+ 1.2 sec (5% overlap on leading edge)
\+ 24.0 sec (time for WRS scene)

+ 1.2 sec (5% overlap on trailing edge)
+ 1.6 sec (378 lines for trailing edge SCA coverage)
= 29.6 sec (29.67 for 7001 lines)

4.1.3.8.1.2 TIRS

The TIRS SCAs are larger and farther apart than the OLI SCAs, which will require TIRS to begin imaging earlier, and continue for a longer duration than OLI in order to image a scene completely. The along-track of the TIRS instrument is 4.95 degrees (Figure 4-16), requiring an additional 9.43 seconds (4.714 seconds for the leading and trailing edges) of imaging to assure all bands in all three SCAs have completely imaged the scene. The same logic used for calculating the number of OLI lines can be used for TIRS.

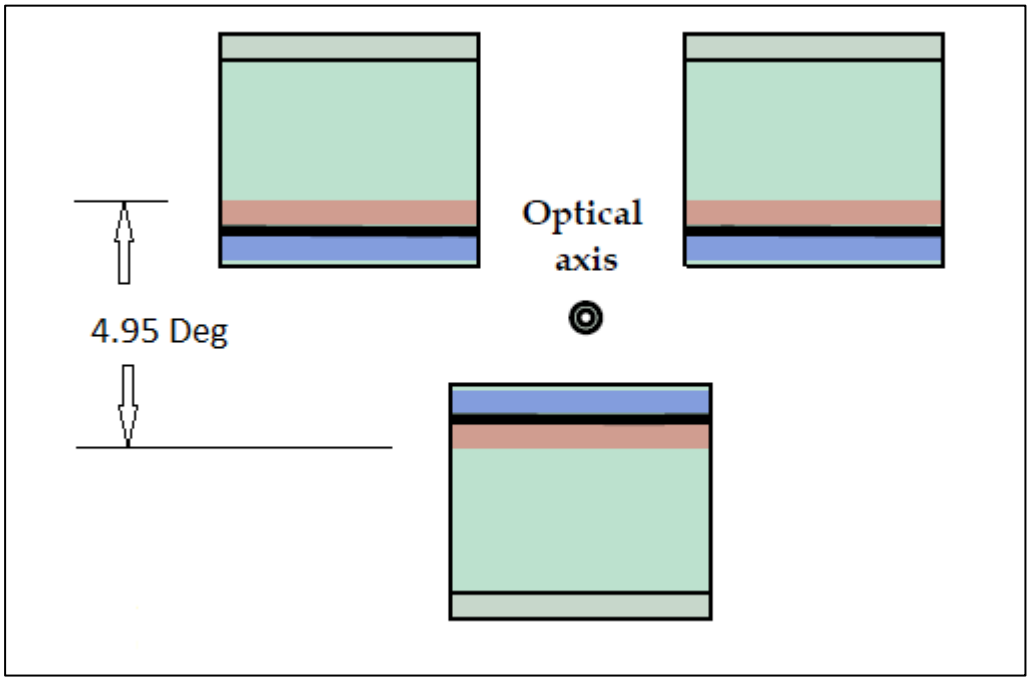

Figure 4-16. TIRS Layout and FOV

The size of a TIRS scene, in lines, can be calculated with respect to the boresight with the following calculations. The number of TIRS lines between successive WRS rows:

TIRS_Frame_Rate = 70 lines / second

$$TIRS\ Lines = \frac{\triangle\, t}{TIRS\_Frame\_Rate} = 1680\ Lines$$

In addition, by definition of Landsat WRS products, WRS scenes are overlapped with the previous row by 5 percent and the next row by 5 percent. The total number of TIRS lines needed for the overlap is then 10 percent.

$$TIRS\ Line = TIRS\ Line + 0.1 * \frac{\triangle t}{TIRS\_Frame\_Rate} = 1848\ Lines$$

Looking at the TIRS Legendre LOS polynomials and determining the leading and trailing look vectors, the difference in radians/degrees is as follows:

Leading_LOS = 4.623841e-02 radians = 2.65 degrees
Trailing_LOS = -3.989761e-02 radians = -2.29 degrees

The difference between these two numbers represents a field of view of 4.94 degrees. Rounding this to 5 degrees (2.7 leading and 2.3 trailing), we can determine the amount of time needed to pad either the leading or trailing acquisition of the TIRS WRS scene in terms of time. If the maximum satellite altitude is present at the poles and the minimum satellite altitude is present at the equator, the minimum and maximum number of lines needed in the L0Rp can be calculated. Using the spacecraft angular rotational rate gives a delta time due to the staggering of the SCAs as:

dτ = 4.92 seconds (maximum)

Using the TIRS frame rate, the number of TIRS lines needed for this (maximum) delta time is as follows:

$$\triangle\ Lines\ SCA\ Staggering = 5\ seconds * TIRS\_Frame\_Rate = 350\ Lines$$

This delta is further extended by 50 lines to include the possible use of the secondary, or *science,* rows. The total number of lines needed within a TIRS WRS scene is then

Total Lines = (nominal size + 5% overlap + SCA staggering + 50 science)

$$TIRS\ Lines = TIRS\ Lines + 2 * (\Delta\ Lines\ SCA\ Staggering + 50) = 2648\ Lines$$

In terms of time, a single scene will take about 37.8 seconds:

5.0 sec (350 lines for leading edge SCA coverage)
+ 0.7 sec (50 lines to include science rows)
+ 1.2 sec (5% overlap on leading edge)
+ 24.0 sec (time for WRS scene)
+ 1.2 sec (5% overlap on trailing edge)
+ 0.7 sec (50 lines to include science rows)
+ 5.0 sec (350 lines for trailing edge SCA coverage)
= 37.8 sec

To ensure TIRS fully covers OLI, additional lines should be added to account for any misalignment between the OLI and TIRS boresights (which has a one-second tolerance) and OLI to ACS alignment, any biases present in the pointing of the SSM, etc. In addition, the above calculations use nominal values,

prelaunch information, and rounding liberties that should be taken into consideration.

The TIRS number of lines can also be calculated based on the number of OLI lines to ensure full coverage of the TIRS data with OLI data. The number of TIRS lines can be found by first scaling the OLI number of lines needed to cover the 180km scene (6230 lines) by the ratio of the nominal line sample rates of OLI-to-TIRS:

$$TIRS\ Lines = 6230\ OLI\ Lines * \frac{70\ Hz}{236\ Hz} = 1848\ Lines$$

This number then needs to be adjusted for the TIRS leading and trailing SCA/band staggering. From the above TIRS calculations, this value is 700 lines. Another 112 lines are needed for the alignment tolerances and 100 lines to include the science rows. The total number of lines needed within a TIRS WRS scene that will fully cover the OLI data is then:

$$TIRS\ lines = 1848 + 700 + 100 + 112 = 2760\ Lines$$

For convenience and ease of use, the final number of TIRS lines will be rounded to 2801 lines. An odd number is used so that a center frame is found and the same number of before and after frames (1400) is used to define the entire L0R scene.

This is equivalent to 40.01 seconds of TIRS data to cover the 29.67 seconds of OLI data. This means TIRS imaging will need to start/end approximately 5.2 seconds before/after OLI to ensure adequate coverage.

From the calculations listed above, the size of the OLI will be 7001 lines (multispectral) and the size of the TIRS will be 2801 lines (thermal).

4.1.3.8.2 Comparison of Partial Scene Definitions

LSDS-270 Landsat 7 (L7) Enhanced Thematic Mapper Plus (ETM+) Level 0 Reformatted Archive (L0Ra) Data Format Control Book (DFCB) has the following discussion of partial scenes:

> For a partial scene with more than half a scene length data, the computed "actual" scene center is also expected to happen in the proximity of the nominal WRS scene center. The "actual" scene center for a greater than half a scene length partial scene may also be computed from the available actual Payload Correction Data (PCD) and indexed to actual data in the band file. For a partial scene with less than half a scene length data, the scene center may have to be computed from extrapolated* ephemeris (no actual PCD may be available from the subinterval). The computed "imaginary" scene center for such a partial scene (less than half a scene) is still determined in the proximity of the nominal WRS scene center, but there will not be any actual band data in the subinterval band file for which

to relate the scene center. The computed "imaginary" scene center for a partial scene with less than half a scene length data is indexed to an "imaginary scan" (non-existent scan 0) in the band file.

* For L8/9, extrapolation of ancillary data beyond the existing ancillary/ephemeris was not provided and/or is not possible. The above method will not work for L8/9, so it was decided that the scene center would actually represent the center of the partial (although the implementation currently takes the first or last line in the scene closest to the WRS-2 center. A future release will address this difference).

The current LSDS-524 Landsat Metadata Description Document (LMDD) definition for L8/9 partial scenes is as follows:

FULL = Full WRS scene - the standard 180 x 185km WRS size.
PARTIAL = Partial WRS scene - less than full and includes the scene center, or greater than half and includes the scene center.

The definition provided in Configuration Change Request (CCR) #598 reads:

PROPOSED: .”..less than a full scene that is not covered by overlap."
RATIONALE FOR CHANGE
-provide consistency with heritage Landsat
-have all scene metadata available in the Inventory
-enable future ability to handle ad hoc requests for partial scenes in Subsetter

The current LSDS-270 Landsat 7 (L7) Enhanced Thematic Mapper Plus (ETM+) Level 0 Reformatted Archive (L0Ra) Data Format Control Book (DFCB) definition for partial scenes is as follows:

Full – A full WRS scene product with approximately ten percent overlap is defined as 180km. For OLI, this is 7001 frames (~28.86-meter MS lines) and for TIRS, this is 2801 frames (~86.91-meter lines).
Partial – Considered less than a full scene.
Scene Center - The computed "actual" scene centers are from the image frame closet to the nominal WRS scene center

4.1.4 Ancillary Data Preprocessing Algorithm

4.1.4.1 Background/Introduction

The ancillary data preprocessing algorithm prepares the ancillary data provided by the spacecraft in the wideband data stream for use by subsequent geometric algorithms. This includes quality checking the incoming data to identify and remove outliers, applying scale factors from the CPF to convert the relevant ancillary data fields to engineering units, and processing the spacecraft attitude and ephemeris data to construct consistent attitude and ephemeris time histories

for the dataset. The baseline assumption is that the attitude and position/velocity estimates produced by the spacecraft will be sufficiently accurate to achieve L8/9 geolocation accuracy requirements. If this is the case, only basic quality checking and smoothing operations will be required. For L9 this quality checking will include verifying that the spacecraft time codes are appropriately converted to time since the J2000 epoch.

The Landsat 8 Spacecraft to Ground Interface Control Document (70-P58230P, Rev C) defines the content and structure of the spacecraft ancillary data. A parallel document for Landsat 9 is in draft form as of this writing. The Landsat 8 Space-to-Ground ICD clarified several uncertainties regarding formats, coordinate systems, and sampling rates that required assumptions to be made in earlier versions of this algorithm description. For example, the rate at which the integrated spacecraft attitude estimates are provided was initially undecided. If the integrated spacecraft attitude estimate quaternions had been provided at a lower rate than the SIRU data, those estimates would have required densification using the raw SIRU data and its associated calibration parameters, status flags, and on-board bias, alignment, and scale estimates. Since the degree of smoothing that the SIRU measurements would be subjected to in the on-board attitude filter was unknown prior to ground system implementation, this algorithm assumed that the raw SIRU measurements would be used to ensure that high-frequency content was preserved. The Smooth Euler and SIRU sub-algorithm is used to perform this data blending and to replace quaternions flagged as outliers.

Although the structure and content of the on-board filtered attitude estimate ancillary data for Landsat 9 is expected to be identical to Landsat 8, the supporting attitude sensor data may be quite different. A different model of star tracker (Sodern rather than Goodrich) will be used for L9 and slightly different model of inertial measurement unit (SIRU-L) will used on L9. The star tracker differences do not impact ancillary data preprocessing since this algorithm uses the integrated attitude estimates rather than the raw star tracker data. The SIRU-L outputs and performance are expected to be the same as for L8, with the differences arising from a more limited set of device characterizations being performed by the SIRU vendor in the case of the –L model. For purposes of the current version of this algorithm, we assume that the SIRU will be unchanged for L9. The potential for relatively late decisions such as these attitude sensor selections, which impact the details of the spacecraft ancillary data content was one of the motivations for separating ancillary data preprocessing out as a stand-alone algorithm.

This algorithm was originally intended to support only imaging intervals that would be suitable for Level 1 processing – primarily Earth-view and lunar acquisitions. Subsequently, it was decided that ancillary data preprocessing would be valuable in other cases, particularly solar calibration intervals. Since these intervals tend to be quite short (only a few seconds), some special logic was added to allow processing to proceed under these conditions. These

adjustments, mainly to the SIRU processing logic, are noted in the appropriate locations below.

4.1.4.2 Dependencies

The ancillary data preprocessing algorithm assumes that ancillary data for the full imaging interval, with (nominally) four seconds of extra data before and after the interval, is available to provide the required geometric support data, that a CPF containing the scale factors needed to convert the ancillary data to engineering units is available, and that the quality thresholds needed to detect and remove or repair outliers are provided either in the CPF or as processing parameters.

4.1.4.3 Inputs

The ancillary data preprocessing algorithm and its component sub-algorithms use the inputs listed in the following table. Note that some of these "inputs" are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data; including dataset IDs to provide unique identifiers for data trending).

Algorithm Inputs
ODL File (implementation)
CPF File Name
Relevant CPF contents:
Ancillary data engineering unit conversion factors
SIRU to ACS alignment matrix
SIRU engineering unit conversion factors
Leap Second Table
Ancillary data thresholds and limits
Orbital Radius Limits (nominal and max excursion)
Ephemeris Angular Momentum Limits (nominal and max excursion)
Quaternion normalization outlier threshold (max difference from 1)
Level 0R Data Directory and File Root Name
Relevant Level 0R spacecraft ancillary data contents:
Spacecraft (S/C) time-tagged inertial to body quaternion estimate
S/C time-tagged ephemeris estimate
SIRU sampling delay (latency) estimate (see note #10)
SIRU clock synchronization times – S/C clock
SIRU clock synchronization times – SIRU clock
SIRU time-tagged delta-angles
SIRU status flags
Output Preprocessed Ancillary Data File Name
Acquisition Type (Earth, Lunar, Stellar, Cal) (optional, defaults to Earth)
Trending on/off switch
WRS Path/Row (for trending)
Geometric Work Order Common Characterization ID (for trending)
Work Order ID (for trending)
Satellite Attributes
Offset (in seconds) from the J2000 epoch to the spacecraft clock epoch (0 for L8, 0.184 for L9)

4.1.4.4 Outputs

The following table contains the ancillary data preprocessing algorithm outputs. It is important to note that the algorithm outputs are independent of the details of SIRU/attitude data processing. Nor would the outputs change if any contingency definitive attitude and/or ephemeris capabilities were required. The ability to provide a stable interface at the output of this algorithm is a large part of the motivation for separating out these ancillary data validation and conversion preprocessing operations from the model creation logic.

Preprocessed Ancillary Data
Attitude Data
Attitude data UTC epoch: Year, Day of Year, Seconds of Day
Time from epoch (one per sample, nominally 50 Hz) in seconds
ECI2ACS quaternion (vector: q1, q2, q3, scalar: q4) (one per sample)
ECEF2ACS quaternion (one per sample)
Body rate estimate (roll, pitch, yaw rate) (one per sample) in radians/second
Roll, pitch, yaw estimate (one per sample) in radians
Ephemeris Data
Ephemeris data UTC epoch: Year, Day of Year, Seconds of Day
Time from epoch (one per sample, nominally 1 Hz) in seconds
ECI position estimate (X, Y, Z) (one set per sample) in meters
ECI velocity estimate (Vx, Vy, Vz) (one set per sample) in meters/second
ECEF position estimate (X, Y, Z) (one set per sample) in meters
ECEF velocity estimate (Vx, Vy, Vz) (one set per sample) in meters/second
Trending Data
WRS Path/Row
Acquisition Date/Time
Geometric Common Characterization ID
Work Order ID
Ephemeris data start UTC time (year, day of year, seconds of day)
Number of ephemeris data points
Number of out of limit ephemeris points
Attitude data start UTC time (year, day of year, seconds of day)
Number of attitude data points
Number of out of limit attitude data points

4.1.4.5 Options

Trending On/Off Switch

4.1.4.6 Procedure

The primary tasks performed by the ancillary data preprocessing algorithm include the following:

1. Preprocess the ancillary ephemeris data:
    a. Load the spacecraft ephemeris data from the interval ancillary data stream.

b. Validate the ephemeris points using orbital radius and angular momentum thresholds.

c. Convert the ephemeris time codes from spacecraft time to a UTC time epoch at the first ephemeris data record time.

d. Correct any time jitter in the ephemeris data samples.

e. Repair any bad ephemeris points by interpolation/propagation.

f. Perform a coordinate conversion to provide the ephemeris in both ECEF and ECI of epoch J2000.0 coordinates.

    i. Convert the incoming ECEF ephemeris state vectors to ECI J2000.

    ii. Convert the ECI J2000 state vectors back to ECEF, removing the effects of Earth rotation from the velocity vectors, as described below.

2. Preprocess the ancillary attitude data:

    a. Load the spacecraft attitude data from the interval ancillary data stream.

    b. Validate the quaternion estimates by computing the magnitude of each and comparing it to 1.

    c. Window the attitude data to ensure that the attitude data are completely within the ephemeris data interval.

    d. Convert the attitude time codes from spacecraft time to a UTC time epoch at the first attitude data record time.

    e. Process the raw SIRU data time stamps to compute sample times relative to the spacecraft clock (if SIRU processing required). SIRU processing is suppressed if this process fails due to the lack of a valid SIRU time synchronization event in the ancillary data interval.

    f. Convert the raw SIRU integrated angle counts to angular rates (if SIRU processing is required and not suppressed).

    g. Rotate the SIRU data to the ACS coordinate system (if SIRU processing is required and not suppressed).

    h. Correct the SIRU data for the effects of orbital motion (Earth-view images only, lunar and stellar acquisitions are left in ECI). Only performed if SIRU processing is required and not suppressed.

    i. Convert the quaternions to roll, pitch, and yaw using the ECI ephemeris data.

    j. Filter the SIRU and quaternion-derived roll, pitch, and yaw values to generate an integrated roll, pitch, yaw and roll-rate, pitch-rate, yaw-rate attitude sequence at the full SIRU data rate. Note: This step will only be used if SIRU data processing is required. If SIRU data

processing is not required or is suppressed, attitude estimates flagged as outliers will be replaced by linear interpolation.

k. Convert the roll, pitch, yaw values to ECI quaternions using the ECI ephemeris data.

l. Convert the roll, pitch, yaw values to ECEF quaternions using the ECEF ephemeris data.

3. Create the output ephemeris and attitude dataset containing:

   a. Attitude Data

      i. Attitude interval UTC epoch as Year, Day of Year, Seconds of Day.

      ii. Attitude sample time offsets from the UTC epoch (in seconds) – one per sample. There will nominally be 50 samples per second.

      iii. Body/ACS to ECI quaternions (vector part q1, q2, q3, and scalar part q4) – one set per sample.

      iv. Body/ACS to ECEF quaternions (vector part q1, q2, q3, and scalar part q4) – one set per sample.

      v. Body inertial rotation rates (roll rate, pitch rate, yaw rate) in radians/second – one set per sample.

      vi. Roll, pitch, and yaw in radians – one set per sample.

   b. Ephemeris Data

      i. Ephemeris interval UTC epoch as Year, Day of Year, Seconds of Day.

      ii. Ephemeris sample time offsets from the UTC epoch (in seconds) – one per sample. There will nominally be one sample per second.

      iii. ECI X, Y, Z position in meters – one set per sample.

      iv. ECI X, Y, Z velocity in meters/second – one set per sample.

      v. ECEF X, Y, Z position in meters – one set per sample.

      vi. ECEF X, Y, Z velocity in meters/second – one set per sample. Note that these are actually ECI velocities rotated into the ECEF coordinate system, not true ECEF velocities that would include Earth rotation velocity.

Steps 1.a., 2.a., and 3 above are input/output functions and are not described further here. The remaining steps are described in detail in the sub-algorithms below.

4.1.4.6.1 Convert Spacecraft Time Code to UTC

The convert spacecraft time code to UTC is a general-purpose sub-algorithm that is used by the more specific ancillary data preprocessing sub-algorithms below. Spacecraft time codes are TAI offsets from the spacecraft clock epoch, selected to be a reference time at (L8) or near (L9) the J2000 astronomic epoch. During Ingest processing, the spacecraft time codes are corrected to be seconds from J2000 by adding the spacecraft clock epoch offset from J2000 (stored in the Satellite Attributes). The ancillary data preprocessing algorithm will verify that this correction has been properly applied to account for the possibility of processing externally generated L0R data, for example, provided by an International Cooperator ground station. This will be done by computing the difference between the Ingest-generated L0R times from J2000 and the original spacecraft time codes in the L0R product, and comparing the difference to the spacecraft epoch offset from J2000, from the satellite attributes.

Since the TAI and UTC time systems differ only by leap seconds, the conversion from TAI seconds since J2000 to UTC amounts to a leap second correction. The J2000 epoch UTC date/time is hard coded (in a #define statement) to prevent it being changed inadvertently. See Section 4.1.2 above for more explanation of the relevant time systems.

1. Load the leap second table from the CPF. The leap second table is represented as the date that each leap second since 01 January 1972 was declared.
2. Scan the leap second table and compute the number of leap seconds prior to the J2000 epoch.
3. Scan the leap second table and compute the number of leap seconds prior to the current spacecraft date/time. This is done by converting the spacecraft time code (TAI offset from J2000) to UTC (without any leap second correction) and then using the resulting “pseudo-UTC” date to determine the number of leap seconds.
4. Subtract the leap second total for the J2000 epoch (result of step 2) from the leap second total for the time code (result of step 3) to compute the number of leap seconds from the J2000 epoch to the current spacecraft time. The resulting number of leap seconds since J2000 is stored the first time it is computed and used in each subsequent time code to/from UTC conversion operation.
5. Subtract the number of leap seconds since J2000 from the TAI seconds from J2000 derived from the spacecraft time code.
6. Add the adjusted seconds from J2000 to the UTC date/time for the J2000 epoch to yield the UTC date/time corresponding to the spacecraft time code.

4.1.4.6.1.1 ECI to/from ECEF Coordinate Transformation

The transformation from ECI of epoch J2000 (mean equator and equinox of J2000.0) to ECEF (WGS84) coordinates is a time-varying rotation due primarily

to the Earth's rotation, but it also contains more slowly varying terms for precession, astronomic nutation, and polar wander. The ECI-to-ECEF rotation matrix can be expressed as a composite of these transformations:

$\mathbf{T}_{ecef/eci} = \mathbf{A\ B\ C\ D}$

A = polar motion
B = sidereal time
C = astronomic nutation
D = precession

Polar Motion
The polar wander correction performs the transformation from the Earth's true spin axis (in the Terrestrial Intermediate Reference System) to the mean pole (in the International Terrestrial Reference System, or ITRS, here taken to be identical to WGS84). The polar motion corrections are tabulated in the CPF. The corrections for the current day are looked up from the CPF table and applied as described in Section 6.5.2 of: Kaplan, George H., United States Naval Observatory Circular No. 179, "The IAU Resolutions on Astronomical Reference Systems, Time Scales, and Earth Rotation Models - Explanation and Implementation," U.S. Naval Observatory, Washington, D.C., October 20, 2005. This document will henceforth be referred to as Circular 179. This transformation is implemented using the *wobble* function in the NOVAS C3.1 library provided by the Naval Observatory.

Sidereal Time
The sidereal time correction performs the transformation from the inertial true-of-date system (true equator and equinox of date) to the Earth-fixed true-of-date (true pole or terrestrial intermediate reference) system. It applies the polar rotation due to GAST, as described in Circular 179. We use the "Equinox-Based" approach described in the Circular and implemented in the *sidereal_time* function of NOVAS C3.1. Note that the sidereal time computation includes the time correction from UTC to UT1 for the current day. The "current day" would be defined by the dataset UTC epoch (rather than being evaluated for each ephemeris or attitude point) to avoid the possibility of introducing leap seconds in the middle of an imaging interval. This correction is tabulated in the CPF, along with the polar wander corrections.

Nutation
The nutation correction performs the transformation from the inertial mean-of-date system (mean equator and equinox of date) to the inertial true-of-date system through nutation angles. The nutation model is based on the IAU 2000 theory of nutation, as described in Circular 179 and implemented in the *nutation* function of NOVAS C3.1.

Precession

The precession correction transforms the inertial of epoch J2000.0 system to the inertial mean-of-date system. The precession model is based on the IAU 2000 definition, as described in Circular 179 and implemented in the *precession* function of NOVAS C3.1. Note that we do not apply the (small) frame bias correction defined in Circular 179 because our target inertial coordinate system is the inertial system of epoch J2000 (ECIJ2000.0) rather than the Geocentric Celestial Reference System (GCRS) described in the Circular.

This transformation rotates a vector from the ECI J2000.0 system to the Earth-fixed (WGS84) system. For example, an ECIJ2000 position vector is converted to ECEF as follows:

$\underline{X}_{ecef} = \mathbf{T}_{ecef/eci}\ \underline{X}_{eci} = \mathbf{A\ B\ C\ D}\ \underline{X}_{eci}$

When working with ephemeris state vectors containing both position and velocity terms, there can be ambiguity in the treatment of the velocity terms when converting between Earth-fixed and inertial coordinates. This ambiguity arises because the transformation is itself time varying. Taking the time derivative of the equation above yields the following:

$$\dot{X}_{ecef} = T_{ecef/eci}\,\dot{X}_{eci} + \dot{T}_{ecef/eci}\,X_{eci}$$

The second term on the right-hand side of the equation captures the time-varying effect of the transformation itself. The time-varying effects of precession, nutation, and polar motion transformations are negligible when compared to the orbital motion of a spacecraft, but the sidereal time transformation contributes a significant effect. Keeping this in mind and expanding $\mathbf{T}_{ecef/eci}$ above yields the following:

$$\dot{X}_{ecef} = A\ B\ C\ D\ \dot{X}_{eci} + A\ \dot{B}\ C\ D\ X_{eci}$$

Where the $\dot{B}$ matrix is defined as the following:

$$\dot{B} = \begin{bmatrix} -\omega^* \sin(GAST) & \omega^* \cos(GAST) & 0 \\ -\omega^* \cos(GAST) & -\omega^* \sin(GAST) & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

With: ω^* = Earth rotation rate in processing reference frame
GAST = Greenwich apparent sidereal time

For useful figures and additional explanation of this transformation, please reference DMA TR8350.2-A, “Supplement to the Department of Defense World Geodetic System of 1984 Technical Report – Part I: Methods, Techniques, and Data Used in WGS 84 Development,” Defense Mapping Agency (now National Geospatial Intelligence Agency), 1 December 1987.

This equation shows that the ECEF velocity is composed of the ECI velocity rotated into the ECEF coordinate system (the first term) plus the effect of Earth rotation (the second term). Note that Earth rotation is modeled by the rate of change of the sidereal time transformation (B) applied to the (ECI true-of-date) position vector.

Whether or not the second term (Earth rotation) is included in the ECI to ECEF velocity transformation depends upon the intended purpose. The original ECEF ephemeris data received from the spacecraft contains velocity estimates that include the Earth rotation effects (i.e., "true" ECEF state vectors). The Earth rotation term must therefore be taken into account when converting these state vectors to ECI J2000. This coordinate system conversion is used to accomplish step 1.f.i above. For most applications within the geometric model, however, we are only interested in the velocity vector as a direction in inertial space (e.g., when using position and velocity to define the orbital coordinate system, which is the attitude control system reference). In this case, we only want the first term – the inertial velocity rotated to ECEF coordinates. Therefore, we rotate the ECI J2000 ephemeris state vectors back to "pseudo" ECEF coordinates without including the Earth rotation term.

To summarize, the ECI/ECEF coordinate system transformations applied to the incoming ephemeris data from the ancillary data file are as follows:

ECEF to ECI:

$$\mathbf{X}_{\text{eci}} = \mathbf{D}^T \mathbf{C}^T \mathbf{B}^T \mathbf{A}^T \mathbf{X}_{ecef}$$

$$\dot{X}_{eci} = D^T\ C^T\ B^T\ A^T\ \left(\dot{X}_{ecef} - A\ \dot{B}\ C\ D\ X_{eci}\right)$$

Noting that the **A**, **B**, **C**, and **D** matrices are orthogonal so that their inverses are equal to their transposes.

ECI to (pseudo) ECEF:

$$\mathbf{X}_{\text{ecef}} = \mathbf{ABCDX}_{eci}$$

$$\dot{X}'_{ecef} = A\ B\ C\ D\ \dot{X}_{eci}$$

Noting that the position vector is the same as the original value, but the velocity vector is not, as indicated by the prime notation.

4.1.4.6.2 Correct Ephemeris Sub-Algorithm

The correct ephemeris sub-algorithm performs steps 1.b., 1.c., and 1.d. of the procedure outlined in Section 4.1.4.6 above. This sub-algorithm quality checks the ephemeris data and corrects any timing jitter errors in the ephemeris solution. The ephemeris values are used to calculate satellite position in the WGS84 ECEF frame.

a) Extract the ephemeris data records from the ancillary data.

b) Search the ancillary data ephemeris records and find the first and last valid ephemeris records in the interval. Extract the time tags for these records.

c) Set the ephemeris epoch to the time associated with the start index found in step b) converted to UTC (see Convert Spacecraft Time Code to UTC sub-algorithm above). Retain the corresponding epoch spacecraft time, as it will be subtracted from the other ephemeris samples.

d) Loop on ephemeris, starting at and ending at indexes found in b.

1) Set ephemeris sample time to the time code from ancillary data minus the ephemeris start time code, i.e., convert times to offsets from the ephemeris epoch defined in c).

2) Convert the ECEF ephemeris position and velocity vectors to ECI J2000 so that the angular momentum check, and subsequent ephemeris smoothing algorithms, can operate in inertial space.

3) Get angular momentum and orbital radius nominal values and allowable deviation thresholds from the CPF: angmo_nom, angmo_delta, orbrad_nom, orbrad_delta.

4) Calculate orbital radius to compare against threshold:

radius = | $\underline{p}$ |

Where: $\underline{p}$ = ephemeris position vector

5) Calculate angular momentum of ephemeris to compare against threshold:

$$\text{angular momentum} = \left\| \vec{p} \times \vec{v} \right\|$$

where:

p = satellite positional vector

v = satellite velocity vector

6) Check orbital radius and angular momentum against nominal values and thresholds from CPF:

If | radius – orbrad_nom | <= orbrad_delta AND
| angular momentum – angmo_nom | <= angmo_delta
Then store the ephemeris point for processing.
Otherwise, report the bad ephemeris point as an outlier.

If fewer than four (the minimum required to support Lagrange interpolation) valid ephemeris points are found, a fatal error is returned for Earth-view, lunar, and stellar acquisitions. For solar calibration acquisitions, additional ephemeris points are propagated using the process model described in the Smooth Position and Velocity Sub-Algorithm below, until four points are available.

The ECI ephemeris is re-interpolated, using the following method, to remove any small time jitters that are present.

Let vx, vy, and vz be the measured velocity.
Let px, py, and pz be the measured position.

e) Loop through the ephemeris points (i = 0 to N-1), computing the distances between adjacent points:

If first ephemeris point (i = 0) set $d_0 = 0$.

If ephemeris point is not first value, then

1) Calculate difference in ephemeris from point i and i-1

$$dx_i = px_i - px_{i-1}$$
$$dy_i = py_i - py_{i-1}$$
$$dz_i = pz_i - pz_{i-1}$$

2) Calculate magnitudes of the delta position and the velocity vectors

$$s_i = \text{sqrt}(dx_i * dx_i + dy_i * dy_i + dz_i * dz_i)$$
$$sv_i = \text{sqrt}(vx_i * vx_i + vy_i * vy_i + vz_i * vz_i)$$

3) Calculate difference between the "predicted time" from the magnitudes calculated in step a2 and the time measured difference between ephemeris points i+1 and i

Set $d_i = s_i / sv_i$ – ephemeris $time_i$ + ephemeris $time_{i-1}$

f) Calculate average difference of time differences from a.

Let $avg = \dfrac{\sum_{i=1}^{N-1} d_i}{N-1}$

Where N = number of ephemeris points

g) Loop through the ephemeris points, adjusting times by the "predicted time difference" (from step e3 above) and the average of "predicted time difference" (from step f above).

ephemeris $time_i$ = ephemeris $time_i + d_i - avg$

Using Lagrange interpolation, calculate satellite position and velocity at one-second intervals, correcting any sampling timing irregularities and filling in any missing outlier points. The time-adjusted satellite position and velocity from the previous step are taken as input.

h) Loop on ephemeris values
   1) Calculate ephemeris interpolation time for current interval (multiple of one second).
   2) Convert ephemeris time from seconds to year, day of year, and seconds of day.
   3) Bracket ephemeris data for interpolation (4 valid points are needed)
      a. Use 2 points before and 2 after the interpolation time.
      b. If that would require points beyond the beginning or end of the ephemeris interval, use the first four or the last four points in the interval.
   4) Interpolate ephemeris to current time (h1) using Lagrange interpolation and bracketed values (h3).

Use the Smooth Position and Velocity sub-algorithm (see below) to smooth the ECI ephemeris. It is then converted to ECEF so that the ECI and ECEF ephemeris representations are consistent (step 1.f.ii. of the procedure outlined in Section 4.1.4.6 above).

4.1.5 Smooth Position and Velocity Sub-Algorithm

A Kalman smoothing filter is used to smooth the ECI position and velocity vectors to accomplish step 1.e. of the procedure described in Section 4.1.4.6 above. The Kalman filter assumes a random process that can be modeled as follows:

$$[X]_{k+1} = [\phi]_k [X]_k + [q]_k + [f]_k$$

where:

$[X]_K$ = (n x 1) state vector at time t_K
$[\phi]_K$ = (n x n) matrix relating X_K to X_{K+1}
$[q]_K$ = (n x 1) process noise at time t_K
$[f]_k$ = (n x 1) forcing function at time t_K

Measurements of the process are modeled as follows:

$$[Z]_K = [H]_K [X]_K + [v]_K$$

where:

$[Z]_K$ = (m x 1) measurement vector at time t_k
$[H]_K$ = (m x n) relates the state vector at time t_k to the measurement
$[v]_K$ = (m x 1) measurement error at time t_k

As noted below, in this application, the measurements are direct observations of the state vector, so n = m.

To smooth the ephemeris data, the state vector X is defined as follows:

$$[X] = \begin{bmatrix} Xp \\ Yp \\ Zp \\ Xv \\ Yv \\ Zv \end{bmatrix}$$

where:

Xp, Yp, Zp = X, Y, Z position
Xv, Yv, Zv = X, Y, Z velocity

The measurement vector [Z] is a 6x1 vector containing the telemetry X, Y, Z positional values along with the telemetry X, Y, Z velocity values. The measurement vector looks like the state vector but contains the measured ephemeris telemetry values for time $t_{k,}$ whereas the state vector contains our estimate of the "true" ephemeris position and velocity values.

The discrete state transition matrix is defined as follows:

$$[\phi]_k = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_k$$

where Δt is the time transition between measurement k and k+1.

The matrix [H] is defined as a 6x6 identity matrix since the measurements directly correspond to the elements of the state vector.

The forcing function, [*f*], is equal to the change in acceleration of the satellite due to the Earth's gravitational potential. The Potential Functions sub-algorithm below describes the forcing function in more detail.

The process noise vector $[q]_K$ represents a random forcing function that models the uncertainty in the dynamic model as a zero-mean random process with covariance [Q]. The process noise controls how strictly the filtered states will conform to the dynamic model.

The process noise covariance matrix is defined as follows:

$$[Q]=\begin{bmatrix} dt^2*\sigma_x{}^2+\frac{dt^4*\sigma_{xv}{}^2}{4} & 0 & 0 & \frac{dt^4*\sigma_{xv}{}^2}{4} & 0 & 0 \\ 0 & dt^2*\sigma_y{}^2+\frac{dt^4*\sigma_{yv}{}^2}{4} & 0 & 0 & \frac{dt^4*\sigma_{yv}{}^2}{4} & 0 \\ 0 & 0 & dt^2*\sigma_z{}^2+\frac{dt^4*\sigma_{zv}{}^2}{4} & 0 & 0 & \frac{dt^4*\sigma_{zv}{}^2}{4} \\ \frac{dt^3\sigma_{xv}{}^2}{2} & 0 & 0 & dt^2*\sigma_{xv}{}^2 & 0 & 0 \\ 0 & \frac{dt^3\sigma_{yv}{}^2}{2} & 0 & 0 & dt^2*\sigma_{yv}{}^2 & 0 \\ 0 & 0 & \frac{dt^3\sigma_{zv}{}^2}{2} & 0 & 0 & dt^2*\sigma_{zv}{}^2 \end{bmatrix}$$

where:

dt=Δt
σ_x=standard deviation in X positional element/value
σ_y=standard deviation in Y positional element/value
σ_z=standard deviation in Z positional element/value
σ_{xv}=standard deviation in X velocity element/value
σ_{yv}=standard deviation in Y velocity element/value
σ_{zv}=standard deviation in Z velocity element/value

The measurement error matrix is 6x6 diagonal matrix:

$$[R]=\begin{bmatrix} m_p & 0 & 0 & 0 & 0 & 0 \\ 0 & m_p & 0 & 0 & 0 & 0 \\ 0 & 0 & m_p & 0 & 0 & 0 \\ 0 & 0 & 0 & m_v & 0 & 0 \\ 0 & 0 & 0 & 0 & m_v & 0 \\ 0 & 0 & 0 & 0 & 0 & m_v \end{bmatrix}$$

where:

m_p = variance of error in positional measurement
m_v = variance of error in velocity measurement

The Kalman filter is used to produce a set of filtered and predicted state vectors, along with estimated and predicted covariance state error matrices. These values are then used to produce a smoothed state vector. This smoothed state vector will represent the smoothed position and velocity ephemeris data.

Prediction equations:

$$[X]_K^p=[\phi]_{K-1}[X]_{K-1}+[f]_{k-1}$$

$$[P]_K^p = [\phi]_{K-1}[P]_{K-1}[\phi]_{K-1}^T + [Q]_{K-1}$$

Filter equations:

$$[K]_K = [P]_K^P[H]_K^T([H]_K[P]_K^P[H]_K^T + [R]_K)^{-1}$$
$$[X]_K = [X]_K^p + [K]_K([Z]_K - [H]_K[X]_K^p)$$
$$[P]_K = ([I] - [K]_K[H]_K)[P]_K^p$$

where:

[*I*] is the identity matrix
[*P*] is the error covariance matrix
[*Q*] = E[$q_t q_t$]
[R] = E[$v_t v_t$]
$[X]^P_K$ = estimate of [*X*] given measurements through t_{K-1}
$[P]^P_K$ = error covariance associated with estimate $[X]_K$
$[X]_K$ = filtered estimate at t_K
$[P]_K$ = filtered estimate at t_K

Note that the filtering step is skipped for points flagged as outliers so that:

$[X]_K = [X]^P_K$
$[P]_K = [P]^P_K$

Using the definitions above, a new notation can be written:

$[X]_{K|K-1} = [X]^P_K$
$[P]_{K|K-1} = [P]^P_K$
$[X]_{K|K} = [X]_K$
$[P]_{K|K} = [P]_K$

The smoothing equations are then:

For n=number of points-1,…,0

$$[X]_{K|N} = [X]_{K|K} + [A]_K([X]_{K+1|N} - [X]_{K+1|K})$$
$$[A]_K = [P]_{K|K}[\phi]_{K+1,K}^T[P]_{K+1|K}^{-1}$$
$$[P]_{K|N} = [P]_{K|K} + [A]_K([P]_{K+1|N} - [P]_{K+1|K})[A]_K^T$$

The Kalman filter is initialized with a state vector:

$$[X]_0 = \begin{bmatrix} Px(0) \\ Py(0) \\ Pz(0) \\ Vx(0) \\ Vy(0) \\ Vz(0) \end{bmatrix}$$

where:

Px(0) = first available X positional value
Py(0) = first available Y positional value
Pz(0) = first available Z positional value
Vx(0) = first available X velocity value
Vy(0) = first available Y velocity value
Vz(0) = first available Z velocity value

The initial error covariance matrix is defined as follows:

$$[P]_0 = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_y^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_z^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_{xv}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_{yv}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{zv}^2 \end{bmatrix}$$

where:

σ_x = initial standard deviation in X position
σ_y = initial standard deviation in Y position
σ_z = initial standard deviation in Z position
σ_{xv} = initial standard deviation in X velocity
σ_{yv} = initial standard deviation in Y velocity
σ_{zv} = initial standard deviation in Z velocity

- Initialize the state vector, error covariance matrix, and measurement error matrix

- Loop on ephemeris points
  - Calculate Δt (time difference between sample time i+1 and i)
  - Calculate process noise matrix
  - Calculate Kalman gain
  - Filter state vector and error covariance matrix
  - Predict error covariance error matrix
  - Calculate force (acceleration) of Earth's mass
  - Predict state

- Initialize Δt to nominal delta time (1 sec)

- Loop on ephemeris (reverse order for smoothing)
  - Calculate smoothed gain
  - Calculate smoothed state
  - Calculate Δt (time difference between sample time i+1 and i)

The resulting $[X]_{K|N}$ are the smoothed ephemeris state vectors.

4.1.5.1 Gravitational Potential Functions

This sub-algorithm calculates the gravitational potential of the Earth, represented as acceleration (x,y,z). One way to model the Earth's gravitational potential is by:

$$\varphi = \frac{\mu}{r}\left[1 - \sum_{n=2}^{\infty} J_n \frac{r_e}{r} P_n \sin(L)\right]$$

where:

J_n = Spherical Harmonics determined by experimentation
μ = Earth's gravitational parameter
r_e = equatorial radius of Earth
P_n = Legendre Polynomials
L = geocentric latitude
sin(L) = z/r

Taking the partial derivatives of the potential function with respect to x, y, and z gives the forcing functions needed for each axis.

$$\frac{\delta\varphi}{\delta x} = \left(-\frac{\mu x}{r^3}\right)\left(1 - 3\frac{J_2}{2}\left(\frac{r_e}{r}\right)^2\left(\frac{5z^2}{r^2} - 1\right) + 5\frac{J_3}{2}\left(\frac{r_e}{r}\right)^3\left(\frac{-7z^3}{r^3} + \frac{3z}{r}\right) - 5\frac{J_4}{8}\left(\frac{r_e}{r}\right)^4\left(\frac{63z^4}{r^4} - \frac{42z}{r^r} + 3\right)\right.$$
$$\left. - 3\frac{J_5}{8}\left(\frac{r_e}{r}\right)^5\left(\frac{231z^5}{r^5} - \frac{210z^3}{r^3} + \frac{35z}{r}\right) + \frac{J_6}{16}\left(\frac{r_e}{r}\right)^6\left(\frac{-3003z^6}{r^6} + \frac{3465z^4}{r^4} - \frac{945z^2}{r^2} + 35\right)\right)$$

$$\left(-\frac{3\mu r_e^2}{r^4}\right)\left(C_{21}\frac{z}{r} - 5C_{21}\frac{x^2 z}{r^3} - 5S_{21}\frac{xyz}{r^3} + 2C_{21}\frac{x}{r} - 5C_{22}\frac{x\left(x^2+y^2\right)}{r^3} + 2S_{22}\frac{y}{r} - 10S_{22}\frac{x^2 y}{r^3}\right)$$

$$\frac{\delta\varphi}{\delta y} = \left(-\frac{\mu y}{r^3}\right)\left(1 - 3\frac{J_2}{2}\left(\frac{r_e}{r}\right)^2\left(\frac{5z^2}{r^2} - 1\right) + 5\frac{J_3}{2}\left(\frac{r_e}{r}\right)^3\left(\frac{-7z^3}{r^3} + \frac{3z}{r}\right) - 5\frac{J_4}{8}\left(\frac{r_e}{r}\right)^4\left(\frac{63z^4}{r^4} - \frac{42z}{r^r} + 3\right)\right.$$
$$\left. - 3\frac{J_5}{8}\left(\frac{r_e}{r}\right)^5\left(\frac{231z^5}{r^5} - \frac{210z^3}{r^3} + \frac{35z}{r}\right) + \frac{J_6}{16}\left(\frac{r_e}{r}\right)^6\left(\frac{-3003z^6}{r^6} + \frac{3465z^4}{r^4} - \frac{945z^2}{r^2} + 35\right)\right)$$

$$\left(-\frac{3\mu r_e^2}{r^4}\right)\left(-5C_{21}\frac{xyz}{r^3} + S_{21}\frac{z}{r} - 5S_{21}\frac{y^2 z}{r^3} - 2C_{22}\frac{y}{r} - 5C_{22}\frac{y\left(x^2+y^2\right)}{r^3} + 2S_{22}\frac{x}{r} - 10S_{22}\frac{y^2 x}{r^3}\right)$$

$$\frac{\delta\varphi}{\delta z} = \left(-\frac{\mu z}{r^3}\right)(1 + 3\frac{J_2}{2}\left(\frac{r_e}{r}\right)^2\left(3 - \frac{5z^2}{r^2}\right) + \frac{J_3}{2}\left(\frac{r_e}{r}\right)^3\left(\frac{30z}{r} - \frac{35z^3}{r^3} - \frac{3r}{z}\right)$$
$$- \frac{J_4}{8}\left(\frac{r_e}{r}\right)^4\left(\frac{-70z^2}{r^2} + \frac{63z^4}{r^4} + 15\right) - \frac{J_5}{8}\left(\frac{r_e}{r}\right)^5\left(\frac{-945z^3}{r^3} + \frac{693z^5}{r^5} + \frac{315z}{r} - \frac{15r}{z}\right)$$
$$+ \frac{J_6}{16}\left(\frac{r_e}{r}\right)^6\left(\frac{4851z^4}{r^4} - \frac{3003z^6}{r^6} - \frac{2205z^2}{r^2} + 245\right))$$
$$\left(-\frac{3\mu r_e^2}{r^4}\right)(C_{21}\frac{x}{r} - 5C_{21}\frac{xz^2}{r^3} + S_{21}\frac{y}{r} - 5S_{21}\frac{yz^2}{r^3} - 5C_{22}\frac{z\left(x^2 - y^2\right)}{r^3} - 10S_{22}\frac{xyz}{r^3})$$

The heritage implementation uses the following six functions to invoke the potential calculations:

p1x - first derivative of X (velocity)
p1y - first derivative of Y (velocity)
p1z - first derivative of Z (velocity)
p2x - second derivative of X (acceleration)
p2y - second derivative of Y (acceleration)
p2z - second derivative of Z (acceleration)

These functions are used to populate the six elements of the forcing function used to Kalman smooth the ephemeris data.

4.1.5.1.1 Attitude Data Preprocessing

This sub-algorithm validates the quaternion attitude estimates and converts their spacecraft time codes to accomplish steps 2.b., 2.c., and 2.d. of the procedure described in Section 4.1.4.6 above.

a) Extract the attitude quaternion data records from the ancillary data.
b) Search the ancillary data attitude records and find the first and last valid attitude records in the interval. Extract the time tags for these records.
c) Adjust the attitude data window as necessary to ensure that it fits entirely within the ephemeris data window.
d) Set the start and stop indexes for the attitude to be stored in the model to the indexes found in step c above.
e) Set the attitude epoch to the time associated with the start index found in step c converted to UTC (see Convert Spacecraft Time Code to UTC sub-algorithm above). Retain the corresponding epoch spacecraft time, as it will be subtracted from the other attitude samples.

f) Loop on attitude records starting at and ending at indexes found in step c.
    a. Set attitude sample time to the spacecraft time code minus the attitude start time, i.e., convert times to offsets from attitude epoch.

b. Compute the magnitude of the attitude quaternion:

Mag = sqrt(q1*q1 + q2*q2 + q3*q3 + q4*q4)

c. Check the magnitude against the nominal value of 1:

a. If the magnitude is between 1-ε and 1+ε, then store the value for processing. The quaternion normalization tolerance value, ε, is nominally 1e-06 (1 part per million) and stored in the CPF.

b. If the magnitude is outside the allowable range, then flag the value as an outlier.

If SIRU processing is required, complete the following steps:

g) Extract the IMU (SIRU) data records from the ancillary data.
h) Process the SIRU clock data to construct spacecraft time codes for each SIRU sample. This is step 2.e. of the procedure described in Section 4.1.4.6 above and is described in the “Process SIRU Time” sub-algorithm below. If this step fails, all subsequent SIRU processing is suppressed by setting a “SIRU_Valid” flag to FALSE.
i) Examine the SIRU status words, flagging any invalid points as outliers.
j) Convert the SIRU counts to angular rates. This is step 2.f. of the procedure described in Section 4.1.4.6 above and is described in the “Process SIRU Counts” sub-algorithm below.
k) Repeat steps b) through e) above for the SIRU data.
l) Use the SIRU epoch as the combined attitude data time epoch. If the SIRU data are not used, the attitude quaternion times are used instead.

4.1.5.1.2 Process SIRU Time Sub-Algorithm

This sub-algorithm analyzes the SIRU clock readouts accompanying each SIRU data sample and uses these in conjunction with the SIRU clock sync offset values and SIRU sync spacecraft time codes included in the Level 0R IMU data records to construct spacecraft time codes for each SIRU data sample.

SIRU sample timing is driven by a clock internal to the SIRU unit. This SIRU clock is a 16-bit counter that increments every 4 microseconds, and rolls over when the 16-bit counter reaches its 64K limit. The SIRU clock is periodically (every 10 seconds or so) synchronized with the spacecraft clock when flight software sends a reset command. Flight software records the spacecraft time code associated with this reset, and the SIRU records the offset between the clock counter value at the time of the reset and the clock counter value at the time of the current data sample in its clock sync field. These offsets are scaled to units of 1/3 of a microsecond (1/12 of the SIRU clock resolution). The clock counter value is not changed by the reset operation, so successfully locating and processing a single reset event is sufficient to establish the timing relationship between the spacecraft and SIRU clocks. The spacecraft time code associated

with the reset is captured in the ancillary data, as is the SIRU sync field. Several additional considerations regarding the SIRU timing data include the following:

1. The SIRU sync field is only populated during the 100 Hz cycle while it is being updated (i.e., while the SIRU is being resynchronized). Otherwise, this field will contain fill. A fill value of -32768 is used for this purpose.
2. One implication of the 100 Hz SIRU cycle is that some sync values will go unrecorded by the 50 Hz L8/9 IMU telemetry. The syncs will be timed such that alternate values will be sampled by and therefore present in the ancillary data. This raises the question of how these unrecorded sync values will be detected and recovered. The SIRU sync spacecraft time code value that accompanies each IMU record will change for the record containing the sync. That would make it possible to use the SIRU sample clock data to recover the missing sync values, though it is probably not important to do so. The sync events that are represented in the data should be sufficient to establish the SIRU/spacecraft timing relationship.
3. Ancillary test data acquired during spacecraft comprehensive performance tests demonstrated that the SIRU clock is not perfectly synchronized to the spacecraft clock. This was manifested as timing jitter in both the SIRU and attitude estimates (which apparently take their times from the corresponding SIRU samples) in which adjacent samples, nominally separated by 0.02 seconds, are sometimes 0.01 and sometimes 0.03 seconds apart, indicating timing drift across the 100 Hz sampling sequence. This has the following effects:
    a. The SIRU rates must be computed using the actual time differences rather than the nominal time difference.
    b. The assignment of times to samples prior to the first SIRU clock sync must use the actual SIRU clock values, not the nominal timing offset.
    c. The SIRU clock syncs are not always visible every 20 seconds (1000 samples apart). The separation is sometimes 500 samples and sometimes 1500 samples due to 100 Hz sampling cycle slippage. This (partly) motivated the inclusion of logic to validate SIRU clock sync events against the previously established timing, to ensure that timing gaps are not introduced into the SIRU data. It also means that intervals shorter than 30 seconds may not contain any valid SIRU clock sync events, leading to the failure of this algorithm and the suppression of further SIRU processing for the interval. This is unlikely in normal (Earth-view, lunar, and stellar) acquisitions, but likely in solar calibration data.
4. The scaling and use of the SIRU latency estimate telemetry in the spacecraft ancillary data stream is not entirely clear. Ancillary datasets from the spacecraft Comprehensive Performance Test (CPT) indicate that the latency is the time offset, in seconds, between the SIRU (and 50 Hz quaternion) data and the flight software 1 Hz cycle times (i.e., the times at which ephemeris and attitude filter outputs are generated). Since this

offset is reflected in the time codes that accompany these data elements, the latency estimates are somewhat redundant. The baseline algorithm does not apply a latency correction.

Three items of SIRU timing telemetry will be used in the following algorithm: the SIRU clock value at the sample time (one per SIRU sample), the SIRU sync reference field (one per sample time, but only valid during resynchronization cycles), and the SIRU sync spacecraft time code (one per IMU record). These will be referred to as clock value (clock), clock sync (sync), and sync time code (time), respectively.

For each IMU record:

Compute the spacecraft time of last sync from the time code seconds and microseconds: time = seconds + microseconds/1e6

For each SIRU sample (i):

If the clock sync field is not fill:

a. Record the current sample clock and time (above) values as base_clock and base_time. Set base_sync equal to base_clock.
b. Compute the SIRU sync offset from the SIRU sync word using the 1/3 microsecond per count scaling factor. The sync word scaling is represented as a ratio relative to the SIRU clock scaling (12 sync counts per clock count):
   sync_time = SIRU sync word * siru_time_scale / SIRU sync ratio
c. Add the SIRU sync offset to the base_time. This the time (in seconds from spacecraft epoch) corresponding to the base_clock SIRU clock value.
d. Initialize the current offset, 16-bit rollover counter and previous offset value:
   siru_offset = 0
   excess_offset = 0
   last_offset = siru_offset
e. Compute the time of the current sample:
   gtime[i] = base_time + siru_offset * siru_time_scale (from CPF)
f. If this is the first valid (non-fill) value, calculate all previous times by working back through the previous SIRU clock samples, subtracting each from the previous time:
   for (j = i to 1)
      Δclock = MOD(siru_clock[j]–siru_clock[j-1]+64K, 64K)
      gtime[j-1] = gtime[j] – Δclock*siru_time_scale
g. Make sure the new clock sync is consistent with previous time codes:
   If abs(gtime[i]-gtime[i-1]-0.02) > 0.02

gtime[i] = gtime[i-1] + 0.02
base_time = gtime[i] – siru_offset*siru_time_scale

This coarse test ensures that the sync update does not introduce a timing adjustment of more than a full 0.02 second sample time. A warning message is generated if this adjustment is made. This test ensures that the SIRU time codes are consistent, and at a minimum, are based on the first sync time in the interval.

Otherwise:

If no valid sync fields have been found, go to the next point.

Otherwise:

Note: All arithmetic involving clock and sync variables is modulo 64K.

Check for a sync that was not sampled:

i. If time > base_time + 0.02 (a resync occurred) AND time < gtime[i-1] + 0.02 (it occurred before this sample):
   1. Reconstruct the sync time:
      sync_time = gtime[i-1] + 0.02 - time
   2. Update the base_time:
      base_time = time + sync_time
   3. Reset the other sync cycle variables:
      base_clock = clock
      base_sync = clock
      siru_offset = 0
      excess_offset = 0

ii. Otherwise:
   1. Calculate the sample time offset based on the current sync variables:
      siru_offset = clock – base_sync (modulo 64K)
   2. Correct for previous 16-bit rollover:
      siru_offset += excess_offset
   3. See if 16-bit rollover occurred on this sample and increment rollover and offset variables; if so:
      if (siru_offset < last_offset)
          excess_offset += 0x010000
          siru_offset += 0x010000

iii. Compute the time of the current sample:
   gtime[i] = base_time + siru_offset * siru_time_scale

iv. Set the last offset value to the current value (used to detect missed sync resets):
   last_offset = siru_offset

If no valid sync values were detected, return an error.

This procedure performs step 2.e.of the procedure described in Section 4.1.4.6 above.

4.1.5.1.3 Process SIRU Counts Sub-Algorithm

This sub-algorithm converts the raw SIRU data counts to angular rates.

For each SIRU sample:

1. If the sample's SIRU validity flags are not set:
    a. Mark the point as an outlier.
    b. If this is the first point, set the angular rates to zero.
    c. Otherwise, set the angular rates to the previous sample values.
2. For valid SIRU samples:
    a. If this is not the first point, compute the difference between the current integrated angle reading and the previous reading for each of the 4 SIRU axes.
    b. If this is the first point, compute the difference between the next integrated angle reading and the current reading for each of the 4 SIRU axes. If the next point is invalid, mark the current point as an outlier and set the angular rates to zero.
    c. Check for SIRU reset/rollover on each axis:
        i. If the value of the angle difference is > 32K, subtract 64K.
        ii. If the value of the angle difference is < -32K, add 64K.
    d. Scale the counts to radians using the SIRU scale factor from the CPF.
    e. The SIRU delta angle measurements are converted to rates by dividing by the delta time computed from the SIRU sample time codes. This could also be done after the four SIRU axis measurements are converted to roll-pitch-yaw measurements using the method described in the next section.

This procedure performs step 2.f. of the procedure described in Section 4.1.4.6 above. For lunar and stellar intervals, all SIRU samples are flagged as outliers so they will be deweighted by the attitude Kalman filter.

4.1.5.1.4 Rotate SIRU Sub-Algorithm

This sub-algorithm rotates the SIRU data to the ACS coordinate frame to accomplish step 2.g. of the procedure described in Section 4.1.4.6 above. Note that this sub-algorithm will only be used if SIRU data processing is required.

Construct the roll, pitch, and yaw rotational matrices from SIRU measurements and rotate angles to the ACS coordinate system. Note that rather than reporting roll, pitch, and yaw rotations directly, the SIRU reports rotations about four non-orthogonal axes oriented in an octahedral tetrad. These four correlated measurements must first be reduced to rotations about the three orthogonal X-Y-Z axes using the SIRU axis vectors from the CPF. These vectors define the orientations of the four SIRU axes, and are nominally:

SIRU A: [+0.57735027 +0.57735027 +0.57735027]
SIRU B: [+0.57735027 -0.57735027 +0.57735027]

SIRU C: [-0.57735027 -0.57735027 +0.57735027]
SIRU D: [-0.57735027 +0.57735027 +0.57735027]

Any one of these SIRU axes can be lost and the rotations about the X-Y-Z axes can still be recovered. The vector for a failed SIRU axis should be set to zero.

Convert SIRU angles to roll-pitch-yaw:

Construct the [S] matrix where the columns of this 3 by 4 matrix contain the four SIRU axis vectors:

$$[S] = \begin{bmatrix} SIRUAx & SIRUBx & SIRUCx & SIRUDx \\ SIRUAy & SIRUBy & SIRUCy & SIRUDy \\ SIRUAz & SIRUBz & SIRUCz & SIRUDz \end{bmatrix}$$

Construct the 3 by 4 SIRU to roll-pitch-yaw conversion matrix:

$$[SIRU2RPY] = \left([S][S]^T\right)^{-1}[S]$$

Construct the 4 by 1 SIRU observation vector:

$$[SIRUOBS] = \begin{bmatrix} \tan(\frac{\theta_A}{2}) \\ \tan(\frac{\theta_B}{2}) \\ \tan(\frac{\theta_C}{2}) \\ \tan(\frac{\theta_D}{2}) \end{bmatrix}$$

where, θ_n = angle for SIRU axis n

Convert the four SIRU observations to three roll-pitch-yaw angles in the SIRU coordinate system:

$$[RPYOBS] = \begin{bmatrix} \tan(\frac{roll}{2}) \\ \tan(\frac{pitch}{2}) \\ \tan(\frac{yaw}{2}) \end{bmatrix} = [SIRU2RPY][SIRUOBS]$$

$$[RPYANG] = \begin{bmatrix} roll \\ pitch \\ yaw \end{bmatrix} = \begin{bmatrix} 2 * a\tan(RPYOBS_1) \\ 2 * a\tan(RPYOBS_2) \\ 2 * a\tan(RPYOBS_3) \end{bmatrix}$$

Construct the perturbation matrix using the SIRU roll-pitch-yaw angles:

[perturbation] = [IRU to ACS] [yaw] [pitch] [roll] [ACS to IRU]

Where [IRU to ACS] is the SIRU to attitude matrix found in the CPF and [ACS to IRU] is its inverse.

The SIRU measured roll, pitch, and yaw in ACS coordinates are then:

$$\text{roll} = -tan^{-1}(\text{perturbation}_{2,1}/\text{perturbation}_{2,2})$$

$$\text{pitch} = sin^{-1}(\text{perturbation}_{2,0})$$

$$\text{yaw} = -tan^{-1}(\text{perturbation}_{1,0}/\text{perturbation}_{0,0})$$

4.1.5.1.5 Correct for Orbital Motion in SIRU Data Sub-Algorithm

The spacecraft SIRU senses rotations relative to inertial space, so it will measure the orbital pitch used to maintain spacecraft pointing, as well as any deviations from that nominal alignment with the orbital coordinate system. In using the SIRU data to densify or repair the spacecraft attitude estimates, they are blended in the orbital coordinate system. Before this can be done, it is necessary to correct the SIRU data for the time-varying orientation of the orbital frame relative to inertial space. This is step 2.h. of the procedure described in Section 4.1.4.6 above.

To account for off-nadir viewing and yaw steering effects, a reference attitude vector representing the mean roll-pitch-yaw for the scene is also provided. This is necessary because the orbital pitch effect will show up in more than just the spacecraft pitch axis if the spacecraft body is not aligned with the orbital coordinate system. The reference attitude is calculated as the average of the roll-pitch-yaw values derived from the quaternion data for the attitude interval.

Note that this sub-algorithm will only be used if SIRU data processing is required.

If the acquisition type is Earth-viewing, then:

Loop on all SIRU values:

1. Calculate the ECI position and velocity of satellite at time t_0 using Lagrange interpolation. The l8_movesat unit does this. That sub-algorithm is included in the LOS Model Creation algorithm.
2. Calculate the transformation matrix from the satellite orbit system to the spacecraft body/ACS coordinate system (ORB2ACS) using the input reference mean attitude. This is the transpose of the ACS2ORB matrix shown in the “Convert Roll, Pitch, Yaw to Quaternion” section below.

3. Calculate the transformation matrix from ECI to satellite orbit system for time t_0 and t_n (the inverse of the ORB2ECI matrix presented in the next section).

   Using the satellite position and velocity at times t_0 and t_n, the following matrix transformations can be calculated:

$$\left[\text{eci2orb}\right]_{time=t0} \Rightarrow \left[\text{eci2orb}\right]_{t0}$$
$$\left[\text{eci2orb}\right]_{time=tn} \Rightarrow \left[\text{eci2orb}\right]_{tn}$$

   Calculate the transformation from the Orbit system to ECI for time t_n using the ephemeris state vector at time t_n.
4. Use the ORB2ACS matrix to compute the ECI2ACS matrices from the ECI2ORB matrices:

$$[\text{eci2acs}]_{t0} = [\text{orb2acs}]\ [\text{eci2orb}]_{t0}$$
$$[\text{eci2acs}]_{tn} = [\text{orb2acs}]\ [\text{eci2orb}]_{tn}$$

   Since the eci2acs matrix is orthogonal, acs2eci can be calculated as:

$$\left[\text{acs2eci}\right] = \left[\text{eci2acs}\right]^T$$

5. Calculate the amount of roll, pitch, and yaw due to the satellite's orbit.

   The roll, pitch, and yaw due to the orbital motion of the satellite can be found by looking at the matrix transformation from spacecraft frame reference at time t_n to spacecraft frame reference t_0.

$$\left[\text{acs}_n\text{2acs}_0\right] = \left[\text{eci2acs}_0\right]\left[\text{acs}_n\text{2eci}\right]$$

$$\Delta\text{roll} = -\frac{\tan^{-1}\left(\dfrac{\left(\text{acs}_n\text{2acs}_0\right)_{2,1}}{\left(\text{acs}_n\text{2acs}_0\right)_{2,2}}\right)}{\Delta\text{time}}$$

$$\Delta\text{pitch} = \frac{\sin^{-1}\left(\left(\text{acs}_n\text{2acs}_0\right)_{2,0}\right)}{\Delta\text{time}}$$

$$\Delta yaw = -\frac{\tan^{-1}\left(\frac{(acs_n\, 2acs_0)_{1,0}}{(acs_n\, 2acs_0)_{0,0}}\right)}{\Delta time}$$

$\Delta time = t_n - t_0$

This formulation computes the orbital attitude rate correction and assumes that the SIRU data are, or have been converted to, rates.

6. Remove orbital motion attitude delta from original values.

$$roll = -roll - \Delta roll$$
$$pitch = -pitch - \Delta pitch$$
$$yaw = -yaw - \Delta yaw$$

The sign is swapped to convert the SIRU angles/rates from body-to-orbit to orbit-to-body.

4.1.5.1.6 Convert to Spacecraft Roll, Pitch, and Yaw Sub-Algorithm

The attitude data are given as quaternions in the ECI reference frame (ECI2ACS). The quaternions are converted to roll, pitch, and yaw values in the ACS reference frame per step 2.i. of the procedure described in Section 4.1.4.6 above.

We first take the conjugate of the incoming ECI2ACS quaternion (**q**) to calculate the corresponding ACS2ECI quaternion (**q'**).

$q'_1 = -q_1$
$q'_2 = -q_2$
$q'_3 = -q_3$
$q'_4 = q_4$

The direction cosines (transformation) matrix from the ACS reference axis to the ECI reference system (ACS2ECI) is constructed from the ACS2ECI quaternion, **q'**, as:

$$ACS2ECI = \begin{bmatrix} q'_1{}^2 - q'_2{}^2 - q'_3{}^2 + q'_4{}^2 & 2(q'_1 q'_2 + q'_3 q'_4) & 2(q'_1 q'_3 - q'_2 q'_4) \\ 2(q'_1 q'_2 - q'_3 q'_4) & -q'_1{}^2 + q'_2{}^2 - q'_3{}^2 + q'_4{}^2 & 2(q'_2 q'_3 + q'_1 q'_4) \\ 2(q'_1 q'_3 + q'_2 q'_4) & 2(q'_2 q'_3 - q'_1 q'_4) & -q'_1{}^2 - q'_2{}^2 + q'_3{}^2 + q'_4{}^2 \end{bmatrix}$$

The ACS2ECI transformation matrix can also be defined as the product of the inverse of the spacecraft's attitude perturbation matrix **P** and the transformation matrix from the orbital reference system to the ECI reference system (ORB2ECI)

The relationship between the orbital and ECI coordinate systems is based on the spacecraft's instantaneous ECI position and velocity vectors. The rotation matrix to convert from orbital to ECI can be constructed by forming the orbital coordinate system axes in ECI coordinates:

$$\vec{n} = -\frac{\vec{p}}{\left\|\vec{p}\right\|}$$

$$\vec{h} = \frac{\left(\vec{n} \times \vec{v}\right)}{\left\|\vec{n} \times \vec{v}\right\|}$$

$$\vec{cv} = \vec{h} \times \vec{n}$$

$$\left[\text{ORB2ECI}\right] = \left[\begin{array}{ccc} \vec{cv} & \vec{h} & \vec{n} \end{array}\right]$$

where:

p = spacecraft position vector in ECI
v = spacecraft velocity vector in ECI
n = nadir vector direction
h = negative of angular momentum vector direction
cv = circular velocity vector direction
[ORB2ECI] = rotation matrix from orbital to ECI

The transformation from orbital to ECI coordinates is the inverse of the ECI to orbital transformation matrix. Since the ECI to orbital matrix is orthogonal, the inverse is also equal to the transpose of the matrix.

$$\left[\text{ORB2ECI}\right] = \left[\text{ECI2ORB}\right]^{-1} = \left[\text{ECI2ORB}\right]^{T}$$

ACS2ECI = **[**ORB2ECI**][P $^{-1}$]**

The orbital reference system to ECI matrix must be defined at the same time as **the** spacecraft's attitude matrix.

The roll, pitch, and yaw values are contained in the $\mathbf{P}^{-1}$ matrix; thus:

$\mathbf{P}^{-1}$ = **[**ORB2ECI**]** $^{-1}$**[**ACS2ECI**]**

The spacecraft attitude is then:

$$\text{pitch} = \sin^{-1}\left(P^{-1}_{2,0}\right)$$

$$\text{roll} = -\tan^{-1}\left(\frac{P^{-1}_{2,1}}{P^{-1}_{2,2}}\right)$$

$$\text{yaw} = -\tan^{-1}\left(\frac{P^{-1}_{1,0}}{P^{-1}_{0,0}}\right)$$

For lunar and stellar intervals, the rotation to the orbital coordinate system is not performed, so the resulting roll, pitch, and yaw values are relative to the ECI system. An additional check is performed on these roll, pitch, yaw values to make sure that there are no crossings of the +/-π radians boundary, with 2π being added or subtracted as necessary to keep the attitude sequence continuous. This check is performed on all intervals, but is only necessary for lunar/stellar data.

4.1.5.1.7 Smooth Euler and SIRU Sub-Algorithm

A Kalman smoothing filter is used to combine the attitude and SIRU data into one data stream and/or replace attitude estimates flagged as outliers per step 2.j. of the procedure described in Section 4.1.4.6 above. Note that this sub-algorithm will only be used if SIRU data processing is required and if the SIRU data are not suppressed.

Lagrange interpolation is used to synchronize the SIRU and quaternion information at the SIRU sampling interval relative to the attitude epoch time. This is necessary because the quaternion and SIRU data sample times are not necessarily uniformly spaced in the original spacecraft ancillary data. The formulation shown here assumes that the SIRU is reporting attitude rate data rather than integrated angles. Due to the increased potential for noise in rate measurements, an additional step is required to synchronize the SIRU data. Specifically, the SIRU rate measurements are integrated to form angles, the angles are Lagrange interpolated to synchronize the times, then the interpolated angles are converted back to rates. The rate to angle integration is performed as follows (the roll, pitch, and yaw axes are each processed separately using this method):

```
SIRU_angle[0] = SIRU_rate[0]*nominal_SIRU_time
For k = 1 to NSIRU-1
        SIRU_angle[k] = SIRU_angle[k-1]
                        + SIRU_rate[k]*(SIRU_time[k] – SIRU_time[k-1])
```

Performing the time regularizing interpolation in angle space suppresses any rate noise present in the SIRU data. The interpolated angles are turned back into rates, suitable for use in the Kalman smoother, as follows:

For k = NSIRU-1 to 1
SIRU_rate[k] = (SIRU_angle[k] – SIRU_angle[k-1])/nominal_SIRU_time
SIRU_rate[0] = SIRU_angle[0] / nominal_SIRU_time

The state vector is defined as follows:

$$[X] = \begin{bmatrix} \text{attitude} \\ \text{iru} \\ \text{drift} \end{bmatrix}$$

where:
attitude = smoothed attitude state
iru = attitude rate state associated with SIRU
drift = slow linear drift error in IRU

The measurement matrix [Z] is a 2x1 matrix containing the Euler and SIRU attitude data for time t_k.

$$[Z] = \begin{bmatrix} \text{epa}_k \\ \text{iru}_k \end{bmatrix}$$

where:
epa = Euler attitude value at time t_k
iru = SIRU attitude value at time t_k

The state transition matrix is defined as follows:

$$[\phi] = \begin{bmatrix} 1 & \text{dt} & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where:
dt = sample timing of SIRU

The matrix [H] is defined as follows:

$$[H] = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -1 \end{bmatrix}$$

The process noise covariance matrix is defined as follows:

$$[Q] = \begin{bmatrix} \frac{\mathrm{dt}^4 * \sigma_{iru}{}^2}{4} & \frac{\mathrm{dt}^3 * \sigma_{iru}{}^2}{2} & 0 \\ \frac{\mathrm{dt}^3 * \sigma_{iru}{}^2}{2} & \mathrm{dt}^2 * \sigma_{iru}{}^2 & 0 \\ 0 & 0 & \mathrm{dt}^2 * \sigma_{drift}{}^2 \end{bmatrix}$$

where:

σ_{iru}=standard deviation of SIRU process
σ_{drift}=standard deviation of drift process

The measurement noise covariance matrix is defined as a 2x2 diagonal matrix:

$$[R] = \begin{bmatrix} m_{euler} & 0 \\ 0 & m_{iru} \end{bmatrix}$$

where:

m_{euler} = observation standard deviation noise in Euler measurement
m_{iru} = observation standard deviation noise in SIRU measurement

Samples flagged as outliers are deweighted by multiplying the measurement standard deviation by 100 for that point.

Each axis is treated as an independent data stream. The Kalman filter is used to produce a set of filtered and predicted state vectors, along with estimated and predicted covariance state error matrices. These values are then used to produce a smoothed state vector. The smoothed vector attitude will represent an overall satellite attitude, or a combination of the Euler and SIRU measurements.

The Kalman filter has an initial state vector of:

$$[X]_0 = \begin{bmatrix} \mathrm{epa}(0) \\ \mathrm{iru}(0) \\ 0 \end{bmatrix}$$

where:

epa(0) = first measured quaternion

iru(0) = first measured SIRU

The initial covariance error matrix is defined as the following:

$$[P]_0 = \begin{bmatrix} \sigma_{epa}{}^2 & 0 & 0 \\ 0 & \sigma_{iru}{}^2 & 0 \\ 0 & 0 & \sigma_{drift}{}^2 \end{bmatrix}$$

where:

σ_{epa} = initial standard deviation in Euler
σ_{iru} = initial standard deviation in SIRU
σ_{drift} = initial standard deviation in drift

Initialize the state vector, error covariance matrix, measurement error matrix, and dt.

Loop on attitude points

- Calculate process noise matrix
- Calculate Kalman gain
- Filter state vector and error covariance matrix
- Predict error covariance error matrix
- Predict state

Loop on attitude points (reverse order for smoothing)

- Calculate smoothed gain
- Calculate smoothed state

The Kalman filtering machinery used here is the same as described in the Smooth Position and Velocity Sub-Algorithm in Section 4.1.5.

If SIRU data processing is not performed, this sub-algorithm is replaced by a simple attitude outlier replacement algorithm that replaces estimates flagged as outliers above, by linearly interpolating new roll, pitch, and yaw values from the neighboring samples.

4.1.5.1.8 Convert Roll, Pitch, Yaw to Quaternion

The roll pitch and yaw sequences computed above are converted to ECI quaternions and to ECEF quaternions per steps 2.k. and 2.l. of the procedure described in Section 4.1.4.6 above. The conversion algorithm is the same in both cases, the only difference being whether the algorithm is provided with ECI ephemeris data or ECEF ephemeris data. See Note 7 in Section 4.1.5.2 below.

Complete the following steps for each attitude sample:

a) Compute the net roll-pitch-yaw by adding the bias value.
b) Use Lagrange interpolation to compute the ephemeris position and velocity at the time of the roll, pitch, yaw attitude sample.
c) Compute the rotation matrix corresponding to the roll-pitch-yaw values: [ACS2ORB] =

$$\begin{bmatrix} \cos(\mathrm{p})\cos(\mathrm{y}) & \sin(\mathrm{r})\sin(\mathrm{p})\cos(\mathrm{y})+\cos(\mathrm{r})\sin(\mathrm{y}) & \sin(\mathrm{r})\sin(\mathrm{y})-\cos(\mathrm{r})\sin(\mathrm{p})\cos(\mathrm{y}) \\ -\cos(\mathrm{p})\sin(\mathrm{y}) & \cos(\mathrm{r})\cos(\mathrm{y})-\sin(\mathrm{r})\sin(\mathrm{p})\sin(\mathrm{y}) & \cos(\mathrm{r})\sin(\mathrm{p})\sin(\mathrm{y})+\sin(\mathrm{r})\cos(\mathrm{y}) \\ \sin(\mathrm{p}) & -\sin(\mathrm{r})\cos(\mathrm{p}) & \cos(\mathrm{r})\cos(\mathrm{p}) \end{bmatrix}$$

d) Construct the rotation matrix to convert from orbital to ECI/ECEF by forming the orbital coordinate system axes in ECI/ECEF coordinates:

$$\vec{n} = -\frac{\vec{p}}{\left\|\vec{p}\right\|}$$

$$\vec{h} = \frac{\left(\vec{n}\times\vec{v}\right)}{\left\|\vec{n}\times\vec{v}\right\|}$$

$$\vec{cv} = \vec{h}\times\vec{n}$$

$$[\mathrm{ORB2EC}] = \begin{bmatrix} \vec{cv} & \vec{h} & \vec{n} \end{bmatrix}$$

where:

p = spacecraft position vector in ECI/ECEF
v = spacecraft velocity vector in ECI/ECEF
n = nadir vector direction
h = negative of angular momentum vector direction
cv = circular velocity vector direction
[ORB2EC] = rotation matrix from orbital to ECI/ECEF

e) Compute the ACS2EC rotation matrix:

[ACS2EC] = **[**ORB2EC**][**ACS2ORB**]**

f) Construct the corresponding EC2ACS quaternion:

First, noting that the ACS2EC matrix computed above can be expressed in terms of the corresponding quaternion components as the following:

ACS2EC =

$$\begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2 & 2(q_1q_2 + q_3q_4) & 2(q_1q_3 - q_2q_4) \\ 2(q_1q_2 - q_3q_4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 + q_1q_4) \\ 2(q_1q_3 + q_2q_4) & 2(q_2q_3 - q_1q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix}$$

We can derive the following set of equations to compute the quaternion components from the elements of ACS2EC:

1. Compute the four quantities:
   $d_1 = 1 + ACS2EC_{11} – ACS2EC_{22} – ACS2EC_{33}$
   $d_2 = 1 – ACS2EC_{11} + ACS2EC_{22} – ACS2EC_{33}$
   $d_3 = 1 – ACS2EC_{11} – ACS2EC_{22} + ACS2EC_{33}$
   $d_4 = 1 + ACS2EC_{11} + ACS2EC_{22} + ACS2EC_{33}$

2. Find the largest of these four quantities and use the corresponding equations to compute the quaternion:

$$if\,(d_1 > MAX(d_2, d_3, d_4)$$

$$q_1 = \frac{1}{2}\sqrt{(1 + ACS2EC_{11} - ACS2EC_{22} - ACS2EC_{33})} = \frac{1}{2}\sqrt{d_1}$$

$$q_2 = \frac{1}{4q_1}(ACS2EC_{12} + ACS2EC_{21})$$

$$q_3 = \frac{1}{4q_1}(ACS2EC_{13} + ACS2EC_{31})$$

$$q_4 = \frac{1}{4q_1}(ACS2EC_{23} - ACS2EC_{32})$$

$$elseif\,(d_2 > MAX(d_1, d_3, d_4)$$

$$q_2 = \frac{1}{2}\sqrt{(1 - ACS2EC_{11} + ACS2EC_{22} - ACS2EC_{33})} = \frac{1}{2}\sqrt{d_2}$$

$$q_1 = \frac{1}{4q_2}(ACS2EC_{12} + ACS2EC_{21})$$

$$q_3 = \frac{1}{4q_2}(ACS2EC_{23} + ACS2EC_{32})$$

$$q_4 = \frac{1}{4q_2}(ACS2EC_{31} - ACS2EC_{13})$$

$$elseif\,(d_3 \geq MAX(d_1, d_2, d_4)$$

$$q_3 = \frac{1}{2}\sqrt{\left(1 - ACS2EC_{11} - ACS2EC_{22} + ACS2EC_{33}\right)} = \frac{1}{2}\sqrt{d_3}$$

$$q_1 = \frac{1}{4q_3}\left(ACS2EC_{13} + ACS2EC_{31}\right)$$

$$q_2 = \frac{1}{4q_3}\left(ACS2EC_{23} + ACS2EC_{32}\right)$$

$$q_4 = \frac{1}{4q_3}\left(ACS2EC_{12} - ACS2EC_{21}\right)$$

$$else$$

$$q_4 = \frac{1}{2}\sqrt{\left(1 + ACS2EC_{11} + ACS2EC_{22} + ACS2EC_{33}\right)} = \frac{1}{2}\sqrt{d_4}$$

$$q_1 = \frac{1}{4q_4}\left(ACS2EC_{23} - ACS2EC_{32}\right)$$

$$q_2 = \frac{1}{4q_4}\left(ACS2EC_{31} - ACS2EC_{13}\right)$$

$$q_3 = \frac{1}{4q_4}\left(ACS2EC_{12} - ACS2EC_{21}\right)$$

This method avoids the numerical danger of dividing by a small number.

We then take the conjugate of the resulting ACS2EC quaternion, **q**, to yield the output EC2ACS quaternion, **q'**:

$q'_1 = -q_1$
$q'_2 = -q_2$
$q'_3 = -q_3$
$q'_4 = q_4$

4.1.5.2 Notes

Algorithm assumptions and notes, including those embedded in the text above, are as follows:

1. The attitude and position/velocity estimates produced by the spacecraft will be sufficiently accurate to achieve L8/9 geolocation accuracy requirements without definitive processing of the raw attitude sensor and/or GPS data.
2. Ancillary data for the full imaging interval (with 4 seconds of extra data before and after the interval) is available to provide the required geometric support data, a CPF containing the scale factors needed to convert the

ancillary data to engineering units is available, and the quality thresholds needed to detect and remove or repair outliers are provided in the CPF.

3. The spacecraft ancillary data will provide attitude estimates (in the form of ECI-to-body quaternions) at the same rate that it provides SIRU data. The L8 spacecraft attitude estimates embody the full SIRU bandwidth. If this had not been the case, then the SIRU data would be needed to restore the high-frequency information to the sequence of attitude estimates.
4. Spacecraft time codes will be TAI offsets from the spacecraft clock epoch, which is the J2000 epoch for L8 and 0.184 seconds after J2000 for L9. Since TAI and UTC differ only by leap seconds, the conversion to UTC amounts to a leap second correction. The UTC time corresponding to the J2000 epoch is hard coded (in a #define statement) to prevent it from being inadvertently changed.
5. Spacecraft ephemeris data will be provided in ECEF rather than ECI coordinates.
6. Ancillary data will include ephemeris and attitude records that contain time tags/time codes (e.g., seconds and fractions of seconds) that are TAI offsets from the spacecraft clock epoch. For L8, the spacecraft clock epoch was the same as the J2000 astronomic epoch. For L9, Ingest processing will convert the spacecraft clock epoch times to times since the J2000 astronomic epoch. Note that J2000 occurred at January 1, 2000, 11:59:27.816 TAI, which corresponds to January 1, 2000, 11:58:55.816 UTC (ref. LDCM Space to Ground Interface Control Document 70-P58230P Rev B). These times reflect the 32.184-second offset between TAI and TT (the J2000 epoch reference frame) and the 32-second offset between TAI and UTC as of J2000. The TAI-UTC offset at J2000 includes the fixed 10-second TAI-UTC offset as of January 1, 1972, and the 22 accumulated leap seconds between then and J2000.
7. The baseline algorithm retains the heritage roll-pitch-yaw attitude model. At some point in the future, this may be replaced by a reformulated model that uses the quaternion representation directly. The sub-algorithm that converts the roll-pitch-yaw attitude representation to a quaternion may not ultimately be used in such a quaternion-based reformulation of the attitude model, since a part of that reformulation would probably involve directly filtering/smoothing the quaternion sequence rather than working in roll-pitch-yaw coordinates. That said, having the capability, provided by this sub-algorithm, to generate a quaternion attitude data representation that is identical to the roll-pitch-yaw representation would simplify the testing of any future reformulation. Note that by including both roll-pitch-yaw and quaternion representations of the attitude data, the algorithm outputs support either approach.
8. Common mathematical algorithms (e.g., matrix and vector operations, Lagrange interpolation) that can be found in standard references (e.g., Numerical Recipes in C) are cited without being described here.
9. The spacecraft estimate of SIRU latency was a late addition to the spacecraft ancillary data stream. Based on our current understanding of

its meaning, it is not needed for SIRU data processing and is not included in the SIRU processing algorithm as of this writing. Should the need for this parameter be established, it should be a straightforward adjustment to the computed SIRU sample times.

10. The terms “IMU,” “IRU,” and “SIRU” are used interchangeably in this document. Inertial Measurement Unit (IMU) is another name for an Inertial Reference Unit (IRU).
11. The ancillary data preprocessing algorithm includes new features (e.g., SIRU processing) but reuses many heritage components (e.g., coordinate system transformations). Some notable modifications to the heritage logic include the following:

    a) The inertial to Earth-fixed coordinate transformation logic was upgraded to include leap seconds (table in CPF to convert from TAI) and light travel time effects (for LOS projection).
    b) The Landsat heritage algorithm for converting between Earth-fixed and inertial coordinates performs simple rotation from the ECI to the ECEF system (or vice versa) for any input vector. This has the effect of rotating the inertial velocity vector to the ECEF frame without incorporating the Earth rotation effect in the velocity. The GPS-derived L8/9 ECEF ephemeris includes Earth rotation effects. As noted above, a new velocity conversion unit was required to implement the velocity equations shown in figure A.1 of DMA TR8350.2-A:
        i. Position: $\mathbf{X}_{ECEF} = [\ ABCD\ ]\ \mathbf{X}_{ECI}$
        ii. Velocity: $\mathbf{V}_{ECEF} = [\ AB'CD\ ]\ \mathbf{X}_{ECI} + [\ ABCD\ ]\ \mathbf{V}_{ECI}$
        iii. Where: B’ is the time derivative of the B matrix.
    c) The heritage IAU 1980 precession and nutation models were replaced with the NOVAS C3.1 implementation of the IAU 2000 models.

4.1.6 Ground Control Point Correlation Algorithm

4.1.6.1 Background/Introduction

The Ground Control Point (GCP) correlation algorithm applies standard image matching techniques to measure the locations of a set of GCPs, each consisting of positional information and an image chip, within a Level 1GT OLI/TIRS image. For each measured GCP, the correlation status (success or failure) and the location within the image where the GCP was expected and where it was actually measured are reported.

The GCP correlation algorithm will be used in two different contexts in the L8/9 Image Assessment System. It will function as part of the primary Level 1TP (L1TP) product generation process, where it provides control point measurements for use by the OLI LOS Model Correction algorithm in registering the L1TP products to the GLS control base. Note on terminology: although geometrically readjusted and augmented with OLI-derived GCPs, the Landsat global control point library is still referred to as the GLS control base since it was originally extracted from the GLS2000 dataset. The GCP correlation algorithm

will also be used for geometric assessment as a tool for measuring the locations of validation GCPs in processed L1TP imagery. These measurements will be analyzed by the Geometric Accuracy Assessment algorithm for data product characterization purposes. High precision control derived from Digital Orthophoto Quadrangle (DOQ) data will also be used at geometric supersites for instrument and platform characterization and calibration.

The L8/9 GCP correlation algorithm was originally derived from the corresponding Advanced Land Imager (ALI) algorithm used in the Advanced Land Image Assessment System (ALIAS). Its implementation is very similar to the gcpcorrelate application. This is a utility algorithm that is not dependent on sensor architecture.

4.1.6.2 Dependencies

The GCP correlation algorithm assumes that GCPs are available for the ground site and that the Model Creation, LOS Projection and Gridding, and Image Resampling algorithms have been executed to create a systematic terrain-corrected L1GT image for GCP mensuration (for the LOS model correction application). This GCP mensuration image may be either SCA-separated or SCA-combined. In either case, the image must match the GCP image chips with respect to ground sample distance, map projection, and image orientation. As such, the band selection and resolution of the input image will depend on the flow being executed/control source being used. For standard L1TP product generation processing, the GLS control points (SWIR1 band, 30m resolution) will be used, whereas for characterization and calibration flows, the DOQ control points (panchromatic band, 15m resolution) will be used. For L1TP product geometric assessment, GLS GCPs flagged as validation points will be extracted and used. A limited set of thermal (ETM+ band 6, 60m resolution) GCPs will also be available to support contingency TIRS-only accuracy characterization operations. These thermal GCPs will use a source identifier of "TM6."

4.1.6.2.1 GCP Retrieval

The GCP mensuration process relies upon a control point storage, management, and retrieval infrastructure. Though not formally part of the GCP correlation algorithm, the availability of logic to retrieve the GCPs corresponding to a particular L1GT image is a dependency of the algorithm. The Landsat 7 production system was the model for this capability and the source of the original GLS-derived operational GCPs. The GCP retrieval logic queries the GCP repository and requests GCP records based upon the following:

1. Geography - GCPs that fall within the latitude/longitude bounds of the L1GT image being correlated. Note that GCPs meeting this criterion could come from more than one WRS path/row, particularly at high latitudes.
2. GCP Source - As noted above, operational L1TP product generation uses GLS control, while the characterization and calibration operations would use DOQ control. Though not shown as an input to this algorithm in the

table below, which takes the pre-assembled control package created by the GCP retrieval process as an input, this GLS, DOQ or TM6 control source selection would be a work order input. Note that there is no requirement to combine GLS, DOQ, and/or TM6 GCPs in a single control set.

3. GCP Type - GCPs will be flagged as "control" or "validation" points so that a subset of the available GCPs can be withheld from the image correction process to provide an independent basis for accuracy assessment (see note #4 in Section 4.1.6.7). Valid query options are as follows: CONTROL, VALIDATION, and BOTH. The CONTROL GCP type will be requested for all cases except the geometric accuracy assessment operation, which will use VALIDATION points. If a correlation failure occurs, a high-priority scene may be reprocessed using the BOTH option. This could help in cases where cloud cover has limited the set of usable GCPs in a particular scene. Under this scenario, if a scene was deemed as necessary for processing (high priority), for characterization, calibration, or other reasons, this scene would be processed through the IAS using the BOTH option.

The GCP retrieval process would extract the GCPs meeting the specified criteria from the GCP repository and construct a GCP data package/library for input to this algorithm. This data package would include the information presented in the Inputs section and in Table 4-4. This algorithm description does not address the implementation details of how the GCP data fields and image chips are stored, managed, and packaged for delivery.

It is probably worth mentioning that the GCP retrieval query may return no valid GCPs. This will happen if, for example, DOQ or TM6 control is requested for an area where it does not exist, or GLS control is requested for a sea ice area or island/reef where control is not available. In this case, the processing system will have to be able to gracefully handle the lack of control (e.g., treat it as a correlation failure and proceed with systematic processing). This is outside the scope of this algorithm description.

4.1.6.3 Inputs

The GCP correlation algorithm uses the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data; including dataset IDs to provide unique identifiers for data trending).

Algorithm Inputs
ODL File (implementation)
Input GCP library/package name/link
Level 1G Image file name
Band to process
Output GCP measurement file name

Calibration Parameter File name (CPF contains default values for processing parameters)
Options and Parameters
Correlation result fit method (see note 2)
Search window size (line, sample) in pixels
Maximum allowable GCP displacement in pixels
Minimum correlation strength (dimensionless)
Image fill value to ignore in correlation
Predicted GCP location offset (line, sample) in pixels (optional) (see note #5)
GCP library/package contents: (see Table 4-4 for details)
Number of GCPs
For each GCP:
GCP ID
GCP type (GLS, DOQ or TM6) (new)
GCP ground position (lat/lon/proj X/proj Y/height) for each GCP
Location of control point within image chip
Chip parameters (e.g., size, ground sample distance (GSD))
Image chip (see note 1)
Level 1G image contents
Image data
Image metadata (DDR), including the following:
Image dimensions (number of lines and samples)
Map projection
GSD/pixel size
Scene corner coordinates

4.1.6.3.1 CPF Parameters

Parameter	Type	Description
GCP Correlation Window Size	Int	Correlation window size
GCP Minimum Correlation Peak	double	Minimum allowable correlation peak strength
GCP Maximum Displacement	double	Maximum allowable measured offsets
GCP Correlation Fit Method	Int	Correlation subpixel peak fit methodology
GCP Correlation Threshold	double	Threshold of allowable fill values in the correlation window
GCP Correlation Fill Value	double	Fill value to check for in the correlation window

4.1.6.4 Outputs

GCP Measurements (see Table 4-5 for details)
GCP ID
Nominal GCP chip line/sample
GCP ground position (lat/lon/height)
Predicted GCP image line/sample
Measured offset from predicted line/sample
Correlation success flag
Correlation coefficient (new)

4.1.6.5 Options

Correlation Fit Method (only one choice in baseline algorithm).
Note that the control source (GLS, DOQ, or TM6) will be selected by the infrastructure software that queries the control database and constructs the GCP

library data package input to this algorithm. As such, it is not strictly an option within this algorithm, but it is an option that this processing step will select.

4.1.6.6 Procedure

This function correlates GCPs located in reference image chips to a terrain-corrected Level 1G image. Windows are extracted from the L1GT image to do the image correlation. The correlated points are written to the GCP data file for subsequent use in precision correction or product evaluation.

The heritage Landsat 7 implementation used L1G mensuration images that were not terrain corrected. The use of terrain-corrected images reduces the size of the L1G image region that must be searched for a control point match. It also requires that the measured GCP locations are associated with elevations interpolated from the Digital Elevation Model (DEM) used to perform the terrain correction. The LOS Model Correction Algorithm describes this (see 4.2.3).

4.1.6.6.1 Ground Control Point Correlation Algorithm Details

GCPcorrelate performs correlations on GCPs with a Level 1GT image.

4.1.6.6.1.1 Ground Control Points

GCPs and reference imagery are generated from USGS Digital Orthophoto Quadrangles (DOQs). DOQs are designed to meet national map accuracy standards at 1:24,000 scale, which corresponds to a Root Mean Square Error (RMSE) of approximately 6 meters. A mosaic of DOQs is created by subsampling the 1-meter DOQ imagery to match the PAN band at a 15-meter resolution. Multiple DOQs are combined so that the mosaic covers a Landsat Worldwide Reference System (WRS) scene extent. The ground control chip library is generated by extracting 64x64 windows from the DOQ mosaic. Since the DOQ data are only available for the United States, these GCPs cover only U.S. test sites.

Ground control chips are chosen by stepping through the DOQ imagery at evenly spaced line and sample locations. Elevation for the chips are found from DEM and stored in the GCP library. If the DOQs that comprise the mosaic have large radiometric differences, histogram equalization operations may be performed. These histogram operations include histogram matching to a reference dataset or histogram balancing within the mosaic.

GCPs have also been extracted from the Global Land Survey 2000 (GLS2000) dataset. These GCPs serve as the geospatial reference for standard Landsat product generation and will be used for L8/9 standard product generation. The GLS2000 GCPs provide a globally distributed, internally consistent control set.

The global set of GLS chips were extracted from Band 5 (SWIR1) at 30m resolution. A limited set of thermal GCPs will also be extracted from the GLS2000 ETM+ band 6 images using a selected subset of GLS scenes. These

thermal GCPs will be used if TIRS geometric characterization must be performed without reference to OLI data.

4.1.6.6.1.2 GCP Mensuration

Throughout the L8/9 data processing and characterization algorithms, normalized cross correlation is used to measure spatial differences between two image sources. These image sources could be OLI and DOQ, OLI and ETM+, TIRS and ETM+, or OLI and OLI. Image windows are extracted and correlation is performed over the windowed area. The correlation process will only measure linear distortions over the windowed areas. By choosing windows that are well distributed throughout the imagery, non-linear differences between the image sources can be found. Normalized cross correlation will produce a discrete correlation surface (i.e., one correlation value per integer pixel offset location). A sub pixel location associated with the offset is found by fitting a polynomial around a 3x3 area centered on the correlation peak. The polynomial coefficients can be used to solve for the sub pixel correlation peak, taken to be the location of the polynomial function maximum. The normalized cross correlation process helps to reduce any correlation artifacts that may arise from radiometric differences between the two image sources.

If the two image windows of size NxM are defined by f and g, the mensuration steps are as follows:

1. Perform normalized gray scale correlation.

$$R(x,y) = \frac{\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2}\left[\left(f(j,i)-\bar{f}\right)\left(g(x+j,y+i)-\bar{g}\right)\right]}{\left[\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2}\left(f(j,i)-\bar{f}\right)^2 \sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2}\left(g(x+j,y+i)-\bar{g}\right)^2\right]^{1/2}}$$

where:

$$\bar{f} = \frac{1}{(M+1)(N+1)}\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2} f(j,i)$$

and

$$\bar{g} = \frac{1}{(M+1)(N+1)}\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2} g(x+j,y+i)$$

R(x,y) is the gray scale discrete correlation surface.

2. Find the peak of the discrete correlation surface by searching for the integer offset with the largest correlation coefficient.

3. Fit a 2nd order polynomial to a 3x3 area centered on the correlation peak. The polynomial has the form:

$$P(x,y) = a_0 + a_1x + a_2y + a_3xy + a_4x^2 + a_5y^2$$

A least squares fit is performed on the points to solve for the polynomial coefficients.

3a) Set up matrices

$$\underset{9x1}{[Y]} = \underset{9x6}{[X]}\underset{6x1}{[a]}$$

where:

X = correlation locations centered around peak
Y = correlation values corresponding to X locations
a = polynomial coefficients

Note that these matrices take the form:

$$\begin{bmatrix} R(-1,-1) \\ R(-1,0) \\ R(-1,1) \\ R(0,-1 \\ R(0,0) \\ R(0,1) \\ R(1,-1) \\ R(1,0) \\ R(1,1) \end{bmatrix}_{9x1} = \begin{bmatrix} 1 & -1 & -1 & 1 & 1 & 1 \\ 1 & 0 & -1 & 0 & 0 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 \\ 1 & -1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & -1 & 1 & -1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}_{9x6} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix}_{6x1}$$

3b) Solve for polynomial coefficients:

$$[a] = \left([X]^T[X]\right)^{-1}[X]^T[Y]$$

4. Find partial derivatives of polynomial equation in terms of x and y:

$$\frac{\partial}{\partial x}P(x,y) = a_1 + a_3y + 2a_4x$$
$$\frac{\partial}{\partial y}P(x,y) = a_2 + a_3x + 2a_5y$$

5. Set partial equations equal to zero and solve for x and y:

$$\text{x offset} = \frac{2a_1a_5 - a_2a_3}{a_3^2 - 4a_4a_5}$$
$$\text{y offset} = \frac{2a_2a_4 - a_1a_3}{a_3^2 - 4a_4a_5}$$

where:

x offset = subpixel offset in x direction
y offset = subpixel offset in y direction

6. Combine the sub-pixel offset calculated in step 5 to the peak location from step 2 to get the total offset.

The GCP positional information and the measured sub-pixel offset is recorded for each GCP, along with a flag indicating whether the final correlation value passed simple correlation strength and maximum offset thresholds. No statistics-based (e.g., t-distribution) outlier detection is performed by this algorithm.

4.1.6.6.2 Processing Steps

The basic GCP Correlation processing flow consists of the following steps:

1. Read the GCPs and L1GT image.
2. For each GCP:
   - 2.1. Compute the predicted location of the GCP in the L1GT image using the GCP map projection coordinates and any specified predicted offset.
   - 2.2. Extract an image window from the L1GT image at the predicted location.
     - 2.2.1. Make sure the image window contains sufficient non-fill image data.
     - 2.2.2. Make sure the L1GT image and GCP image chip are in the same map projection (UTM zone). Reproject (see below) the GCP chip if necessary.
   - 2.3. Correlate the GCP image chip with the L1GT window to find the optimum match point.
   - 2.4. Test the measured correlation and offset against predefined thresholds.
   - 2.5. Write out the GCP mensuration results.

The reprojection of the GCPs is done as a separate step through the process called gcpretrieve. This process is a precursor to the actual correlation process. It is also worth noting that the resampling methodology is slightly different between the Landsat heritage code and the prototype. The Landsat methodology used a table of weights that were applied to each chip in order to perform the reprojection.

The GCP correlation procedure was implemented in the heritage ALIAS prototype. Though the correlation process is conceptually simple, it is computationally intensive so the ALIAS implementation was designed to be efficient. This included taking advantage of parallel processing. These processing efficiency measures make the heritage implementation somewhat more

complicated than it might otherwise be. The remainder of this processing discussion follows the L8/9 prototype, which was based on the heritage ALIAS implementation, to illustrate how the conceptually simple flow outlined above was mapped to a computationally efficient implementation.

4.1.6.6.3 Prototype Code

Input to the executable is an ODL file; output is an ASCII file containing measured offsets between the input image file and GCP library. Under the prototype output/input file directory, there is a directory called chips that contains the GCP data structures and files. The prototype output/input directory also contains the ODL files needed, the HDF5 input image file, a Perl script that is needed, and the CPF.

The following sections briefly describe the functions of the primary modules of the prototype implementation.

Get GCP Correlate Parameters (get_gcpcorrelate_parameters)
This function gets parameters from the ODL parameter file.

Get GCP Information (get_gcp_information)
This function reads the GCPs from the input GCP library.

Process GCP (process_gcp)
This function processes all of the GCPs by extracting the GCP image chip, extracting the image window, performing the correlation for each point, and then writing the results to the GCP data file.

Notes:

1. The correlation routines want things in sample, line order, so the fit_offset pairs returned are horizontal (sample) and then vertical (line) offsets. In contrast, the GCP data file contains fit_offset in line, sample order.
2. To calculate the correlated location, know the following:
    a. The correlate routines return the offset from the reference window (chip) to the search window (L1GT), which is also the offset from the nominal reference point to the actual point in the search window.
    b. The integer location of the predicted location roughly corresponds to the integer location of the reference location. We need to report the predicted search line, sample of the reference point, and the offset from the predicted point to the correlated point. To get the correlated location, we start with the integer location of the predicted point because this corresponds to the integer part of the ref point (this is why gcp[num_used_gcp].fit_offset subtracts the fractional part of the predicted location). Then, we add the fractional part of the reference coordinate because this is really the

point we are going after. After that, we add the correlation fit_offset because this tells how the reference point relates to the location in the search window.

3. The calculation for fit_offset only works correctly because we are assuming the reference and search points are at the center of the window (plus some fractional distance), and the difference in window size is accounted for by nom_off; if the reference point was not at the middle of the chip, this would have to be adjusted.

Initialize Parallel Correlator (xxx_init_parallel_correlator)
This function initializes an instance of the parallel correlator. All of the multiprocessing resources are created, and the memory for the chip buffers and queue structures is allocated.

Get Correlation Chip Buffers (xxx_get_corr_chip_buffers)
This function returns buffers for the search and reference chips that will be submitted to the parallel correlator. Getting buffers from this routine and not submitting them to the parallel correlator will quickly exhaust all of the buffers available. The buffers will be freed when the parallel correlator is closed. When compiled in single-threaded mode, the same set of buffers is used for every pair of chips.

Close Parallel Correlator (xxx_close_parallel_correlator)
This function is the routine that needs to be called after all of the chips have been submitted to the correlator. This routine will wait until all of the threads have completed, then destroy this instance of the parallel correlator. The results of the correlation are not valid until this routine returns.

Get Search Line/Sample (get_search_line_samp)
This function finds the line and sample that corresponds to the given projection y and x. Since the L1GT image is positioned (map projection) north up, to find the line (sample), subtract the upper-left projection y(x) value from the GCP projection y(x) value and divide the result by the line(sample) pixel size.

Notes:
The line and sample numbers are 0-relative.
This will not work for a path-oriented image.

Extract Window (ias_misc_extract_window)
This function extracts an image window around a specific GCP. From the input image, a window of the specified size will be extracted around the GCP line and sample. If the window is an odd size, the extra line and/or sample will be at the beginning of the imagery. The data in the window representing portions outside of the imagery will be filled with zeros.

The two steps to the extraction are as follows:

1. Data type conversion of whatever the L1GT image is to float
2. Setting the calculated window correctly into the buffer (even if the calculated window falls partially outside of the image)

Check Fill (oli_check_fill)
This function checks the input window for fill data over the specified percentage. This routine is useful to determine if there is too much fill data in a window. If too much fill data exists, then the window might not be good for correlating. Fill data nominally has a value of 0.0.

Extract Chip (xxx_extract_chip)
This function reads the specified image chip. The image chip is always assumed to be a flat binary file containing chip_size[0] x chip_size[1] BYTE pixels (see note #1).

Resample chip if necessary (build_gcp_lib)
New logic, derived from the Landsat Product Generation System (LPGS) heritage, will be required here to check the image chip map projection, and if necessary, resample the chip to match the L1GT image. This is necessary when working with a global GCP repository containing GCPs extracted from multiple source scenes in multiple UTM zones. The GCPs falling inside of a particular L1GT image will frequently (particularly at high northern latitudes) be drawn from source images in neighboring UTM zones. It is also worth noting that the resampling techniques between the LPGS heritage code and the L8/9 prototype are not the same.

Image chip reprojection proceeds as follows:

1. Compute the image chip UL corner coordinates from the GCP UTM coordinates, the GCP image line/sample coordinates, and the image chip pixel size:
    a. UL Corner X = GCP X - GCP sample coordinate * chip pixel size
    b. UL Corner Y = GCP Y + GCP line coordinate * chip pixel size

   Note that the GCP line/sample coordinates are relative to a zero-origin at the center of the upper-left chip pixel.
2. Project the GCP latitude and longitude to the L1GT map projection (UTM zone) using the projection transformation package (see OLI/TIRS LOS Projection Algorithm) to compute GCP X' and GCP Y' projected coordinates.
3. Compute the desired "new" chip UL corner in the L1GT projection using the new GCP X' and GCP Y' coordinates, rounding off to a whole multiple of the pixel size:
    a. UL Corner X' = GCP X' - GCP sample coordinate * chip pixel size
    b. UL Corner Y' = GCP Y' + GCP line coordinate * chip pixel size

c. UL Corner X' = round(UL Corner X'/chip pixel size)*chip pixel size
d. UL Corner Y' = round(UL Corner Y'/chip pixel size)*chip pixel size

4. Compute the "new" GCP line/sample coordinates in the reprojected chip:
   a. GCP sample coordinate' = (GCP X' - UL Corner X')/chip pixel size
   b. GCP line coordinate' = (UL Corner Y' - GCP Y')/chip pixel size
5. For each point in the new chip:
   For line = 0 to nlines-1
   Compute: Y' = UL Corner Y' - line*chip pixel size
   For samp = 0 to nsamps-1
   1. Compute: X' = UL Corner X' + samp*chip pixel size.
   2. Convert (X',Y') to old chip projection (X,Y) using the projection transformation package.
   3. Compute: oline = (UL Corner Y - Y)/chip pixel size.
   4. Compute: osamp = (X - UL Corner X)/chip pixel size.
   5. If the point (oline, osamp) falls inside of the old chip boundary, then interpolate a DN value at that location using bilinear interpolation:
      a. lindex = (int)floor(oline)
      b. sindex = (int)floor(osamp)
      c. fline = oline - lindex
      d. fsamp = osamp - sindex
      e. DN(oline,osamp) =
         DN(lindex,sindex)*(1-fline)*(1-fsamp) +
         DN(lindex+1,sindex)*fline*(1-fsamp) +
         DN(lindex,sindex+1)*(1-fline)*fsamp +
         DN(lindex+1,sindex+1)*fline*fsamp

   Else DN(oline,osamp) = 0
6. Use the reprojected image chip and GCP line/sample coordinates in the GCP correlation procedure.

The build_gcp_lib, or gcpretreive process, is separated from the GCPcorrelate process to emulate the GCP retrieval process from the database containing the GCP image chips and their corresponding metadata. This retrieval process also contains the following C modules:

GCPretrieve - Main driver for GCP retrieval process.

get_gcp_lib - Reads GCPs according to set of criteria

get_gcp_information - Wrapper for reading GCPLib information

get_gcp_proj_parms - Reads projection information from the image file metadata

get_gcpretrieve_parameters - Read input ODL parameters

This code was based on the Landsat ETM+/TM heritage code for GCP retrieval and chip reprojection.

Put GCP (io_put_gcp)
This function writes all GCP records to the specified output file. This function writes out a set of GCP data records. If the file already exists, it will be overwritten.

Write GCP (io_write_gcp)
This function writes one record to the GCP data file. The file pointer is left at the end of the current record so sequential calls of xxx_write_gcp will sequentially write all of the records in the file. The GCP data file is assumed to be an ASCII file containing one line of text per GCP data record. Each record contains: point_id, reference_line, reference_sample, latitude, longitude, elevation, predicted_search_line, predicted_search_sample, delta_y (line), delta_x (sample), accept/reject_flag, correlation coefficient, reference band number, search band number, search SCA number (0 for SCA-combined images), chip source (DOQ, GLS).

Submit Chip to Correlator (xxx_submit_chip_to_corr)
The xxx_parallel_corr module implements a parallel correlation object. Using the Posix threading interface, up to MAX_CORR_THREADS (or the number of processors available - whichever is less) are created to perform correlation. The main thread that creates the parallel correlator is then responsible for "feeding" the parallel correlator chips to correlate. The xxx_submit_chip_to_corr places the chips into a queue. The correlation threads remove the chips from the queue and perform the correlation. The results of the correlation are not immediately available to the application since xxx_submit_chip_to_corr returns before the correlation is complete.

Before any of the correlation results are used, the application must call xxx_close_parallel_correlator to make sure all of the chips have been correlated and to destroy the correlation threads.

Grey Correlator (xxx_grey_corr)
This function correlates a reference subimage with a search subimage using the pixel grey levels, and evaluates the results.

Grey Cross Product Same-size (xxx_grey_cross_ss)
This function computes the unnormalized (raw) sum of pixel-by-pixel cross products between the reference and search images for every combination of horizontal and vertical offsets of the reference relative to the search image for windows of the same size (in one dimension at least).

Grey Normalization Same-size (xxx_grey_norm_ss)
This function converts raw cross-product sums to normalized cross-correlation coefficients, using tabulated statistics from previous step

(grey_cross_ss). This function is much simpler than the one for unequal-sized windows, since all normalizing is done by the space domain same size correlator. All that has to be done is statistics gathering to set up the peak finder.

Grey Cross Product (xxx_grey_cross)
This function computes the unnormalized (raw) sum of pixel-by-pixel cross-products between the reference and search images for every combination of horizontal and vertical offsets of the reference, relative to the search image. This function works for windows of unequal size.

Grey Normalization (xxx_grey_norm)
This function converts raw cross-product sums to normalized cross-correlation coefficients, while tabulating statistics needed for subsequent evaluation. This function works for unequal window sizes.

Grey Evaluation (xxx_grey_eval)
This function evaluates various measures of correlation validity and extracts a subarea of the cross-correlation array centered on the peak.

Fit Registration (xxx_fitreg)
This function fits a quadratic surface to the neighborhood of the correlation peak and from it determines the best-fit registration offsets and their estimated errors.

4.1.6.6.4 Input and Output File Details

The details of the fields contained in the input GCP library file (Table 4-4) and the output measured GCP file (Table 4-5) are as follows:

Field	Description
Header Text	Zero or more lines of ASCII text, each line beginning with the "#" symbol to designate it as a header comment, describing GCP library contents.
Data Start Marker	"BEGIN" - static text to indicate beginning of data area
Number of GCPs	Integer number (N) of GCPs to follow (new)
GCP Record Fields:	One set per GCP
GCP Number	Sequence number of GCP in this package (1 to N)
GCP ID	Unique ID for GCP of the form: ppprrrnnnn where: ppprrr = WRS path/row GCP was taken from nnnn = chip sequence number
GCP Image Chip Line Coordinate	GCP location within image chip - line coordinate (fractional pixel).
GCP Image Chip Sample Coordinate	GCP location within image chip - sample coordinate (fractional pixel).
GCP Latitude	GCP latitude in degrees.
GCP Longitude	GCP longitude in degrees.
GCP X	GCP projected X coordinate in meters.
GCP Y	GCP projected Y coordinate in meters.
GCP Height	GCP WGS84 ellipsoid height in meters.

Image Chip GSD	Chip pixel size in meters.
Image Chip Lines	Number of lines in the image chip.
Image Chip Samples	Number of samples in the image chip.
GCP Source	Source of GCP, either "DOQ," "GLS," or "TM6"
GCP Type	Control/validation point flag, either "CONTROL" or "VALIDATION"
Image Chip Projection	UTM or PS
Image Chip Zone	UTM zone number (1-60). Use 0 for PS.
Image Chip Date	yyyymmdd = year/month/day of the GCP source
Image Chip	Link to chip image data (could be in file named with GCP ID)

Table 4-4. Input GCP Library Contents

Field	Description
GCP Record Fields:	One set per GCP
Point ID	GCP ID (see Table 4-4)
GCP chip line location	Line location of the GCP within the chip
GCP chip sample location	Sample location of the GCP within the chip
GCP latitude	GCP WGS84 latitude in degrees
GCP longitude	GCP WGS84 longitude in degrees
GCP height	GCP WGS84 ellipsoid height in meters
Predicted GCP image line	Predicted line location of the GCP in the L1GT image
Predicted GCP image sample	Predicted sample location of the GCP in the L1GT image
GCP image line offset	Measured line offset from the predicted location
GCP image sample offset	Measured sample offset from the predicted location
Correlation success flag	Flag 0 = correlation failure, 1 = success
Correlation coefficient	Measured correlation coefficient (new)
Search band number	L1GT band number used
Search SCA number	L1GT SCA where the GCP was found
Chip source	GCP source (DOQ or GLS or TM6)

Table 4-5. Output GCP Mensuration File Contents

4.1.6.7 Notes

The following are additional background assumptions and notes:

1) The heritage GLS, TM6, DOQ image chips are stored as 8-bit (BYTE) arrays, whereas the L8/9 imagery is 16-bit. The correlation is performed on floating-point data so both the image and the chips are converted to float on input. Thus, the image and chip data types need not match.
2) The correlation result fit method defines the algorithm used to estimate the correlation peak location to subpixel accuracy. Only the quadratic surface-fitting method described in this description is supported in the baseline algorithm.
3) Though the normal baseline for measuring control points is to use an SCA-separated terrain corrected image, this algorithm should also function with a combined-SCA image so that it can be used to measure test-point GCPs in L1TP product images to support the geometric accuracy characterization algorithm.
4) The GCPs in the GCP repository are flagged as either “control” points, to be used for LOS model correction, or “validation” points, to be used for geometric accuracy characterization. The utility that extracts control points

from this repository must be able to extract either control set. The “control” set would contain the majority of the points. The “validation” flag is only used in areas where more than some minimum threshold number of GCPs are available. The Cal/Val team set these flags when the GCP repository was loaded and could adjust them thereafter, if necessary. Criteria for selecting validation points are based upon considerations such as the following:

a) The total number of available GCPs in the scene must exceed some minimum (e.g., 100).
b) Points that fall on the boundary (or, more precisely, the convex hull) of the GCP set would not be validation point candidates.
c) Points that are within some maximum distance (e.g., 25 km) of another GCP would be validation point candidates.
d) The goal would be to develop an automated validation point identification algorithm that would operate somewhat like an outlier rejection algorithm: identify the best validation point candidate based on a set of criteria, remove it from the control point list, and iterate until no additional validation points are identified.

5) Scenes with poor geolocation accuracy can lead to the actual GCP L1GT image locations being sufficiently far from their predicted locations so as to make it impractical to expand the GCP search window to the extent necessary to find the GCPs. An optional parameter to specify an a priori predicted offset provides a more reliable way to find and correctly correlate the GCPs in this situation. This can occur early in the mission, before the first on-orbit sensor alignment calibration, or during an anomaly investigation.

4.1.7 Solar and View Angle Generation Algorithm

4.1.7.1 Background/Introduction

Landsat 8/9 Level 1TP (and 1GT) products (henceforth referred to as L1T products) provide radiometrically and geometrically corrected geolocated image samples for each spectral band. These samples are 16-bit fixed point numbers that can be related to either at-sensor radiance or reflectance using parameters provided in the product metadata. The L1T samples are also precisely registered to a UTM (or polar stereographic) map projection grid which makes it straightforward to construct pixel ground coordinates from the product corners.

For some applications additional information about the scene geometry is needed, including elevation, slope/aspect, sensor viewing angles (zenith and azimuth), and/or solar illumination angles. This algorithm provides a method for generating per-pixel sensor viewing and solar illumination angles for L1T products by providing an angle coefficient file, containing selected information from the geometric model and resampling grid, and associated logic for using the new file to compute the required angles.

The Operational Land Imager (OLI) and Thermal Infrared Sensor (TIRS) payloads on the Landsat 8/9 mission are both pushbroom imagers with focal planes that span the full Landsat swath width. Full swath coverage is achieved by using multiple sensor chip assemblies (SCAs) across-track, with sufficient overlap between adjacent SCAs to avoid coverage gaps. This SCA-to-SCA overlap is achieved by displacing alternate SCAs along-track so that adjacent SCAs can cover overlapping portions of the across-track field of view. For the OLI, which uses 14 SCAs to cover the full swath, the 7 odd SCAs (1 through 13) are arranged to point slightly forward of nadir and the 7 even SCAs (2 through 14) are arranged to point slightly aft. Similarly for TIRS, the central SCA-C points forward while the outboard SCAs (A and B) point aft. The layout of the OLI focal plane is shown in Figure 4-17 below.

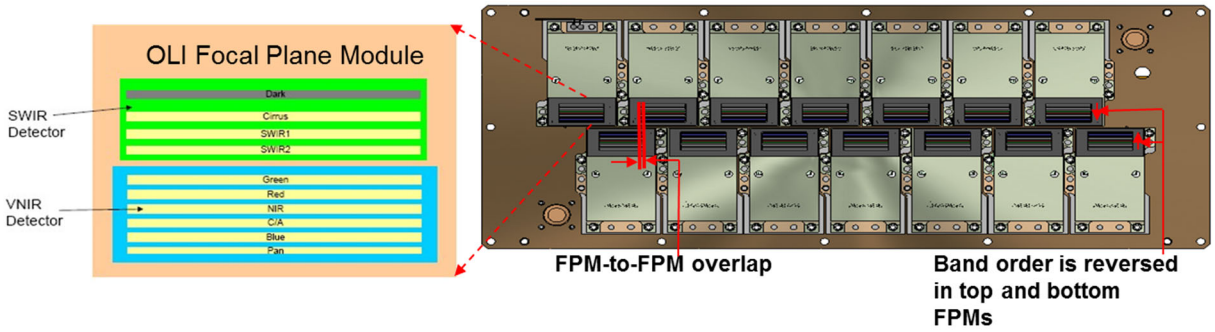

Figure 4-17. OLI Focal Plane Layout

Figure 4-17 raises a problem of terminology. Ball Aerospace uses the term SCA for just the detector and ROIC chips (without filters) and refers to the complete unit as a focal plane module (FPM). The TIRS developers used SCA to refer to the entire assembly, while the FPM acronym had an entirely different meaning. The Cal/Val team decided to adopt SCA as the standard terminology for both instruments.

A key challenge in analyzing the viewing geometry for both the OLI and TIRS sensors is the along-track offset between adjacent SCAs as this focal plane geometry leads to discontinuities in the viewing geometry at SCA boundaries. The view angle changes occasioned by the alternating even/odd SCA geometry would make it difficult to fit a simple function to the not-very-smooth angle patterns. This argues for generating and storing the view angles for each pixel. On the other hand, the along-track distribution of the spectral bands, also shown in Figure 4-17, ensures that the viewing angles will be different for each spectral band. Explicitly representing the angles for each pixel in each band would thus be space prohibitive as the angle file would be larger than the L1T product. These considerations led to a compromise solution, described herein, that uses multiple rational polynomial functions to model the viewing geometry for each band on each SCA. These functions are implemented in an exploitation tool that uses scene-specific parameters, stored in an angle coefficient file provided with each L1T product, to generate viewing angles on demand.

The algorithm is thus implemented in two parts. The first part, intended to run in the IAS/LPGS environment at product generation time, uses the geometric model and grid files used to create the L1T product, to build an additional angle coefficient file that accompanies the product. This new file captures the elements of the scene geometry needed to subsequently calculate the solar illumination and sensor viewing angles for each active (i.e., those that contain OLI or TIRS image data) product pixel. The second part of the algorithm uses the angle coefficient file to compute these angles. This part is implemented as a standalone software tool that is available to the user community (see https://www.usgs.gov/land-resources/nli/landsat/solar-illumination-and-sensor-viewing-angle-coefficient-files for more details). Two versions of this tool were developed: an experimental version that supports multiple processing options and a simpler operational prototype that provides only basic angle generation capability.

4.1.7.2 Dependencies

The angle generation algorithm assumes that the standard L1T geometric modeling algorithms have run successfully and that the geometric model, geometric grid, and calibration parameter files used to create the L1T product are available. The angle computation algorithm can optionally use an input elevation model for improved accuracy. If provided, this model must match the scene frame (corners, projection, pixel size) of the L1T product multispectral bands.

4.1.7.3 Inputs

The solar and view angle generation algorithm and its component sub-algorithms use the inputs listed in the following table.

Algorithm Inputs – Angle Coefficient File Generation
Geometric Grid File
Scene Framing Information:
Scene corner coordinates
Scene map projection information:
Projection GCTP code: 1 = UTM, 6 = Polar Stereographic
UTM zone number (1-60)
GCTP map projection parameters
Datum and spheroid codes (WGS84)
Geometric Model File
WRS path and row (orbital, for use in output file name construction)
Image times
Spacecraft ephemeris (position vs. time)
Calibration Parameter File
Earth model parameters
WGS84 ellipsoid parameters
Earth orientation parameters (UT1-UTC offset, pole wander)
Leap second table
NOVAS solar ephemeris (sun ECITOD direction vs. time)
Algorithm Inputs – Angle Computation

Angle Coefficient File – see output table below for contents
DEM File (optional)
WGS84 ellipsoid height (in meters) for each 30-meter pixel in the L1T product
Subsampling Factor (optional)

4.1.7.4 Outputs

The solar and view angle generation algorithm outputs are shown in the following table. See Table 4-6 below for more detail contents of the output angle coefficient file.

Algorithm Outputs – Angle Coefficient File Generation
Angle Coefficient File
File Header
Angle coefficient file name
Satellite ID
WRS path and row (orbital)
List of bands included
Projection Information
Ellipsoid parameters
Projection type/code
Projection units (meters)
Projection spheroid and datum (WGS84)
UTM zone number
GCTP projection parameters
L1T product projection corners
Ephemeris Data
UTC epoch (year, day of year, seconds of day)
Number of ephemeris points
Time from epoch for each point
ECEF X, Y, and Z position for each point
Solar Vector Data
UTC epoch (year , day of year, seconds of day)
Number of solar vectors provided
Time from epoch for each vector
ECEF X, Y, and Z directions for each vector
Rational Polynomial Coefficient Data for each Band
Number of SCAs
Number of lines and samples in L1T product
Number of lines and samples in L1R input (full scene)
L1T pixel size (in meters)
Image start time relative to ephemeris epoch
Image line time (time between lines)
Mean height in scene
Mean L1R line/sample coordinates in scene
Mean L1T line/sample coordinates in scene
Mean satellite viewing vector components (local east-north-vertical coordinates)
Rational polynomial numerator coefficients for the viewing vector X component
Rational polynomial denominator coefficients for the viewing vector X component
Rational polynomial numerator coefficients for the viewing vector Y component
Rational polynomial denominator coefficients for the viewing vector Y component
Rational polynomial numerator coefficients for the viewing vector Z component
Rational polynomial denominator coefficients for the viewing vector Z component

Mean solar illumination vector components (local east-north-vertical coordinates)
Rational polynomial numerator coefficients for the solar vector X component
Rational polynomial denominator coefficients for the solar vector X component
Rational polynomial numerator coefficients for the solar vector Y component
Rational polynomial denominator coefficients for the solar vector Y component
Rational polynomial numerator coefficients for the solar vector Z component
Rational polynomial denominator coefficients for the solar vector Z component
List of SCAs in current band
Rational Polynomial Coefficient Data for each SCA
Mean height in SCA
Mean L1R line/sample coordinates in SCA
Mean L1T line/sample coordinates in SCA
Rational polynomial numerator coefficients for L1R line coordinate
Rational polynomial denominator coefficients for L1R line coordinate
Rational polynomial numerator coefficients for L1R sample coordinate
Rational polynomial denominator coefficients for L1R sample coordinate
Algorithm Outputs – Angle Computation
Satellite Viewing Angle File for each Band
Viewing zenith angle for each L1T pixel (unless subsampled)
Viewing azimuth angle for each L1T pixel (unless subsampled)
Zenith and azimuth “bands” are sequential
Zenith and azimuth angles are stored as 16-bit integers scaled to units of 0.01 degrees
Satellite Viewing Angle (ENVI) Header File (one per angle file)
Number of lines and samples in angle file
Number of bands in angle file (2)
Data type (signed 16-bit integer)
Interleaving type (BSQ)
Projection information
Projection type (UTM or PS)
UTM zone/PS projection parameters
Output angle file pixel size (in meters) = L1T pixel size * subsampling factor
UL corner coordinates
Solar Angle File for each Band
Solar zenith angle for each L1T pixel (unless subsampled)
Solar azimuth angle for each L1T pixel (unless subsampled)
Zenith and azimuth “bands” are sequential
Zenith and azimuth angles are stored as 16-bit integers scaled to units of 0.01 degrees
Solar Angle (ENVI) Header File (one per angle file)
Number of lines and samples in angle file
Number of bands in angle file (2)
Data type (signed 16-bit integer)
Interleaving type (BSQ)
Projection information
Projection type (UTM or PS)
UTM zone/PS projection parameters
Output angle file pixel size (in meters) = L1T pixel size * subsampling factor
UL corner coordinates

4.1.7.5 Options

Angle computation can work with or without elevation data input.
The output angle “bands” can be optionally subsampled.

4.1.7.6 Procedure

The primary tasks performed by the solar and view angle generation algorithm are to:

1. At product generation time: create an angle coefficient file that contains all of the information a user needs to calculate per-pixel solar illumination and sensor viewing angles for each band in the L1GT/L1TP (L1T) data product.
2. On demand for the user: use the angle coefficient file to generate solar illumination and sensor viewing angles that correspond to the L1GT/L1TP (L1T) product pixels.

Phase 1: Generate Angle Coefficient File

Central to the ability to compute the satellite viewing or solar illumination geometry for a particular L1T image pixel is the ability to associate that pixel with its time of observation. Once the time is known, it can be used to calculate the spacecraft position, from which the sensor viewing geometry is derived, and the solar direction, from which we calculate sun angles. The key to mapping output product image pixels to imaging time is to reconstruct the relationship between the resampled L1T product pixels and the unresampled L1R calibrated detector samples from which they are derived, since there is a simple linear relationship between L1R line number and time. The L1R to L1T mapping can be calculated from the geometric model. To facilitate efficient L1T product generation, this relationship is stored in the geometric grid file for an array of points spanning the image bounds. The goal here is to formulate a set of equations that represent, in a compact form, the input space (Level 1R) line/sample to output space (Level 1T) line/sample mappings contained in the geometric grid file. Experimentation has shown that sub-pixel accuracy in the L1T line/sample to L1R line/sample mapping can be achieved using rational polynomial functions of the following form:

$$L1R_{Line} = L1R_{MeanLine} + \frac{(a_0 + a_1 * L1T_L + a_2 * L1T_S + a_3 * Hgt + a_4 * L1T_L * L1T_S)}{(1 + b_1 * L1T_L + b_2 * L1T_S + b_3 * Hgt + b_4 * L1T_L * L1T_S)} \quad (1)$$

$$L1R_{Sample} = L1R_{MeanSample} + \frac{(c_0 + c_1 * L1T_L + c_2 * L1T_S + c_3 * Hgt + c_4 * L1T_L * L1T_S)}{(1 + d_1 * L1T_L + d_2 * L1T_S + d_3 * Hgt + d_4 * L1T_L * L1T_S)} \quad (2)$$

Where:

$L1T_L = L1T_{Line} - L1T_{MeanLine}$

$L1T_S = L1T_{Sample} - L1T_{MeanSample}$

$Hgt = Height - Height_{Mean}$

a0 to a4, b1 to b4, c0 to c4, and d1 to d4 are model coefficients.

One set of rational polynomial coefficients (RPCs) is computed for each band on each SCA using the information in the geometric grid file. For the OLI, there are thus 23 model parameters (18 polynomial coefficients plus 5 mean offsets) per band/SCA. With 14 SCAs and 9 bands, this results in a total of 2898 model constants per scene. For the TIRS, there are 23 model parameters / band / SCA * 3 SCAs * 2 bands for a total of 138 additional model constants. This is a large but manageable number. Note that five model parameters; the mean values of the input and output coordinates; are added to reference the rational polynomial formulation to the center of the band/SCA area covered by the functions. This helps provide numerical stability in the least squares solution for the model coefficients.

Calculating the Model Coefficients

The geometric grid file contains a set of three-dimensional arrays of L1R to L1T pixel mappings, one for each band on each SCA. The array axes are L1R line, L1R sample, and (ellipsoid) height, with each array point corresponding to one L1R line / L1R sample / height triplet. This is depicted in Figure 4-18. The L1T line / sample location corresponding to each triplet is computed using the line-of-sight projection model and the selected output L1T scene frame, and the results are stored in the grid structure for subsequent use during image resampling. The grid thus provides all of the information required to solve for the rational polynomial model coefficients.

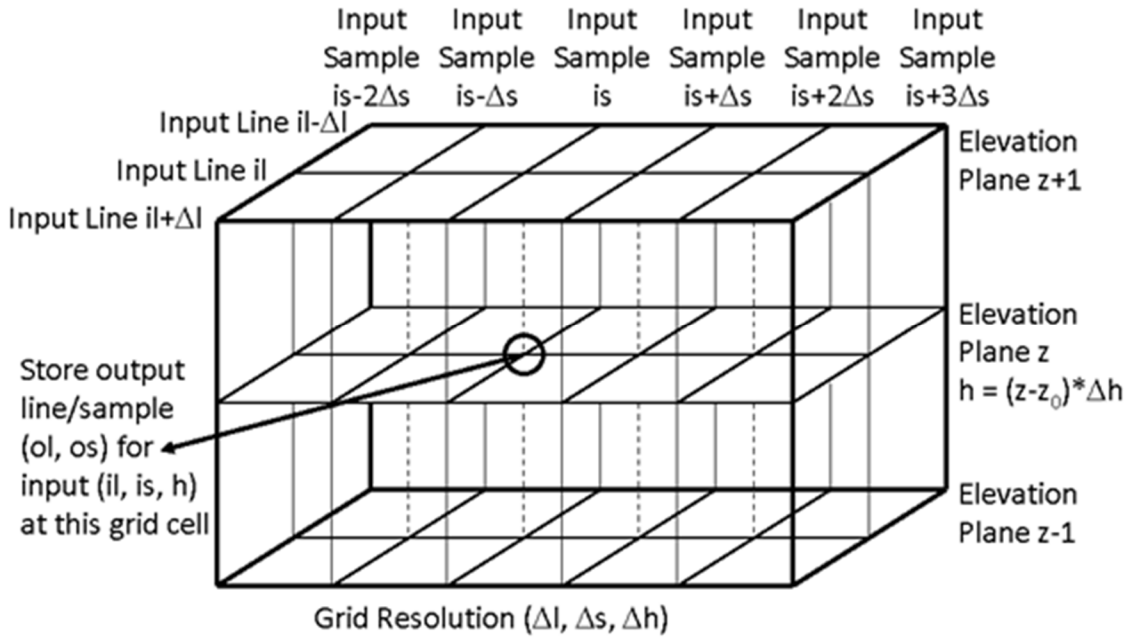

Figure 4-18. Geometric Grid Structure

All of the grid points for a given band/SCA are used to solve for the model coefficients for that band/SCA. Calculating the mean L1R line, L1R sample, L1T

line, L1T sample, and height values is straightforward. The model coefficients are determined by a least squares solution. To accomplish this, the rational functions are linearized by multiplying the denominator by the left hand side and rearranging terms as follows:

$$L1R_L = (a_0 + a_1 * L1T_L + a_2 * L1T_S + a_3 * Hgt + a_4 * L1T_L * L1T_S) - (b_1 * L1T_L * L1R_L + b_2 * L1T_S * L1R_L + b_3 * Hgt * L1R_L + b_4 * L1T_L * L1T_S * L1R_L) \quad (3)$$

$$L1R_S = (c_0 + c_1 * L1T_L + c_2 * L1T_S + c_3 * Hgt + c_4 * L1T_L * L1T_S) - (d_1 * L1T_L * L1R_S + d_2 * L1T_S * L1R_S + d_3 * Hgt * L1R_S + d_4 * L1T_L * L1T_S * L1R_S) \quad (4)$$

Where:

$L1R_L = L1R_{Line} - L1R_{MeanLine}$

$L1R_S = L1R_{Sample} - L1R_{MeanSample}$

One pair of equations of this form can be constructed for each grid point. Standard least squares techniques are used to solve for the nine coefficients in each equation.

Constructing the Angle Coefficient File

The sequence of activities required to assemble the information required to build the angle coefficient file is as follows:

1. Open and read the input data files:
    a. Load the geometric model from the LOS model file.
    b. Load the geometric grid from the grid file.
    c. Load the Earth model parameters from the CPF.
    d. All of these operations are accomplished using standard IAS library input/output modules.
2. Initialize the Earth model:
    a. Initialize the IAS time conversion library functions using the leap second table read from the CPF.
    b. Store the WGS84 ellipsoid semi-major and semi-minor axes from the CPF in the angle coefficient (ANGCF) structure.
3. Get path/row, ephemeris, and sun vector information from the geometric model.
    a. Store the WRS path and row from the model in the ANGCF structure.
    b. Extract ephemeris data covering the current scene from the geometric model.
        i. Using the ephemeris start time (UTC epoch) as a reference, calculate the time offsets to the first and last line in each band by invoking the ias_math_get_time_difference utility.
        ii. Determine the earliest band start time (this will normally be the TIRS bands) and latest band end time.

iii. Subtract 8 seconds from the earliest start time and add 8 seconds to the latest end time to get the target time bounds for extracting ephemeris data to cover the scene.
iv. Find the index of the first ephemeris point with a time after the target start time and the index of the last ephemeris point with a time before the target end time.
v. Establish a new ephemeris epoch at the time of the first sample to be extracted.
vi. Load the time (adjusted for the new epoch) and ECEF position fields for the selected ephemeris points from the model into the ANGCF structure.

c. Use NOVAS to compute ECEF sun vectors at the ephemeris sample times.
i. Initialize the NOVAS solar ephemeris package.
ii. For each ephemeris point:
1. Construct the full UTC time by adding the point's time offset to the ephemeris UTC epoch.
2. Convert the UTC year and day of year to month and day.
3. Use the year, month, day, and seconds of day to compute the Julian day required by NOVAS.
4. Invoke NOVAS to compute the ECI true-of-date solar direction vector at the specified Julian day.
5. Use IAS library routines to convert the ECITOD sun vector to ECI of epoch J2000 (by applying nutation and precession models). Note that the IAS library ECEF/ECI coordinate transformation routines also invoke NOVAS.
6. Use IAS library routines to convert the ECIJ2000 sun vector to ECEF, including the pole wander and UT1-UTC corrections from the geometric model.
7. Load the time and ECEF solar unit vector into the ANGCF structure.
iii. Shut down the NOVAS package.

4. Get map projection and scene corner information from the geometric grid.
a. Load the projection code, units, zone, spheroid, datum, and GCTP map projection parameter fields from the grid into the ANGCF structure. These parameters are needed to convert map X/Y to geodetic latitude/longitude.
b. Load the scene corner map projection coordinates from the grid into the ANGCF structure. The corners are needed to convert L1T line/sample to map projection X/Y.
5. Initialize the IAS library map projection logic.
a. Construct a map projection structure using the parameters loaded in the ANGCF structure, by invoking the ias_geo_set_projection module.

b. Construct a geodetic projection structure to produce latitude/longitude coordinates in radians, using the ias_geo_set_projection module.
c. Construct a projection transformation that converts map X/Y to geodetic latitude/longitude using the structures from a. and b. above, and the ias_geo_create_proj_transformation module.
d. Pre-establishing this transformation will make subsequent map projection conversion computations easier.

6. Assemble the band-specific angle coefficient structure fields for each band.
   a. Load the band number, number of SCAs, number of L1T lines/samples, and pixel size from the grid into the ANGCF structure.
   b. Load the number of L1R lines/samples, band start time, and line increment time (sampling time) from the geometric model into the ANGCF structure.
   c. For each SCA, record the SCA number in the ANGCF structure and calculate the coefficients of the L1T-to-L1R RPC model described above.
      i. Compute the mean height of the grid points as: Mean_hgt = (num_Zplanes – 1 – 2*zeroplane)*Zspacing/2
      ii. Compute the mean L1R line and sample values by cycling through the grid in_lines and in_samps arrays.
      iii. Compute the mean L1T line and sample values by cycling through the grid point out_lines and out_samps arrays.
      iv. Loop through all the points in the grid for this band/SCA to construct the normal equations:
         1. The form of the observations is shown in equations (3) and (4) above. Each grid point yields one line and one sample observation, expressed in matrix/vector notation:

$$X_L^T \theta_L = Y_L \quad X_S^T \theta_S = Y_S$$

Where:

$$X_L = \begin{bmatrix} 1 \\ L1T_L \\ L1T_S \\ Hgt \\ L1T_L * L1T_S \\ -L1R_L * L1T_L \\ -L1R_L * L1T_S \\ -L1R_L * Hgt \\ -L1R_L * L1T_L * L1T_S \end{bmatrix} \quad \theta_L = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix} \quad Y_L = [L1R_L]$$

$$X_S = \begin{bmatrix} 1 \\ L1T_L \\ L1T_S \\ Hgt \\ L1T_L * L1T_S \\ -L1R_S * L1T_L \\ -L1R_S * L1T_S \\ -L1R_S * Hgt \\ -L1R_S * L1T_L * L1T_S \end{bmatrix} \qquad \theta_S = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \\ c_4 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \end{bmatrix} \qquad Y_S = [L1R_S]$$

2. Each observation of each type (line or sample) contributes to the normal equations:
   $N_L\theta_L = L_L$ and $N_S\theta_S = L_S$
   Where we have accumulated the N and L matrices as:
   $N_L \mathrel{+}= X_L X_L^T$ $\qquad L_L \mathrel{+}= X_L Y_L$
   $N_S \mathrel{+}= X_S X_S^T$ $\qquad L_S \mathrel{+}= X_S Y_S$

v. Once the observation contributions from all of the grid points are collected into the normal equation matrices, we solve for the unknown rational polynomial coefficient vectors θ_L and θ_S:
   $\theta_L = N_L^{-1} L_L$ $\qquad \theta_S = N_S^{-1} L_S$

vi. This procedure generates a set of L1T-to-L1R RPCs for each SCA in the band.

The procedure described thus far provides everything necessary to support the computation of the required view and sun angles: a mechanism for relating L1T product line/sample to L1R line/sample which yields time of observation; the ECEF position of the spacecraft as a function of time; the ECEF direction to the sun as a function of time; and the scene framing and projection information needed to convert L1T line/sample to map X/Y, then to geodetic (optionally including height from an input DEM), and finally to ECEF. Though feasible, this approach requires the application of multiple complex coordinate transformations for every pixel in the L1T product. Experiments with this approach to generating view and sun angles demonstrated that, while angle accuracies of 1 arc-minute or better can be achieved if terrain data are included, the required computations are rather time consuming. On the IAS development platform, generating satellite viewing and solar illumination zenith and azimuth angles for every imaged pixel in all bands required 40-45 minutes of processing time. In an effort to reduce this processing time, a more computationally efficient alternative was developed.

Rapid Angle Computation

In the alternate approach, a second-tier rational polynomial model is fitted directly to the satellite and sun unit viewing vectors making it possible to compute them

directly. This circumvents the need for complex map projection and geodetic computations involving trigonometric functions. Unit vector components, rather than the angles themselves, are fitted to avoid the +/-180 degree azimuth discontinuity.

The second-tier "angle" rational polynomial functions, one set per band, are more complicated than the first-tier per-SCA L1T-to-L1R rational functions, because they must account for the SCA-to-SCA discontinuities. This is achieved by including both L1T and L1R input terms in the formulation. The RPC model equation for the satellite viewing unit vector X component is:

$$Sat_X = Sat_{XMean} + \frac{Num_x(L1T_L, L1T_S, Hgt, L1R_L, L1R_S)}{Den_x(L1T_L, L1T_S, Hgt, L1R_L, L1R_S)} \quad (5a)$$

$$\begin{aligned} &Num_x(L1T_L, L1T_S, Hgt, L1R_L, L1R_S) \\ &\quad = a_0 + a_1 * L1T_L + a_2 * L1T_S + a_3 * Hgt + a_4 * L1R_L + a_5 \\ &\quad * L1T_L * L1T_L + a_6 * L1T_L * L1T_S + a_7 * L1T_S * L1T_S + a_8 \\ &\quad * L1R_S * L1R_L * L1R_L + a_9 * L1R_L * L1R_L * L1R_L \end{aligned} \quad (5b)$$

$$\begin{aligned} &Den_x(L1T_L, L1T_S, Hgt, L1R_L, L1R_S) \\ &\quad = 1 + b_1 * L1T_L + b_2 * L1T_S + b_3 * Hgt + b_4 * L1R_L + b_5 * L1T_L \\ &\quad * L1T_L + b_6 * L1T_L * L1T_S + b_7 * L1T_S * L1T_S + b_8 * L1R_S \\ &\quad * L1R_L * L1R_L + b_9 * L1R_L * L1R_L * L1R_L \end{aligned} \quad (5c)$$

Where:

$L1T_L = L1T_{Line} - L1T_{MeanLine}$
$L1T_S = L1T_{Sample} - L1T_{MeanSample}$
$Hgt = Height - Height_{Mean}$
$L1R_L = L1R_{Line} - L1R_{MeanLine}$
$L1R_S = L1R_{Sample} - L1R_{MeanSample}$
a_0 to a_9, and b_1 to b_9, are the RPC model coefficients.

There are similar models for Sat_Y, Sat_Z, Sun_X, Sun_Y, and Sun_Z.

The terms included in these equations were determined by experimentation to minimize the rational polynomial model fit residuals. Note that in order to use these models it is necessary to first evaluate the L1T-to-L1R RPC model to determine the values for $L1R_L$ and $L1R_S$.

The final steps in the assembly of the angle coefficient file are to compute these "angle" rational polynomial model coefficients, using a procedure much like that described in step #6 above, and to write out the angle coefficient ODL file:

7. Calculate the direct angle RPCs for each band.
    a. Calculate the satellite viewing vector and the solar illumination vector in the local vertical coordinate system at each point in the geometric grid.

i. Extract the height (from the grid Z-plane), L1R line/sample, and L1T line/sample for the current point from the grid structure.
ii. Use the L1T corners and pixel size to convert L1T line/sample to map X/Y:
X = upleft_X + $L1T_S$ * pixel_size
Y = upleft_Y – $L1T_L$ * pixel_size
Note that this assumes projection north-up products, which all L8/9 L1T products currently are. This could be made more elaborate to support path-oriented products if necessary since all four scene corners are included in the angle coefficient file.
iii. Use the (already initialized) map projection transformation to convert map X/Y to latitude/longitude using IAS library functions. These functions implement the map projection algorithms documented in, “Map Projections – A Working Manual” by John P. Snyder, USGS Professional Paper 1395, U.S. Government Printing Office, Washington, DC, 1987.
iv. Convert geodetic latitude, longitude, and height (from the grid) to ECEF X, Y, Z using IAS library functions. The details of the geodetic to ECEF transformation are described in the OLI Line-of-Sight Projection/Grid Generation Algorithm Description Document. This yields the ground point ECEF vector $\mathbf{G_{ECEF}}$.
v. Calculate the local vertical coordinate system basis vectors from the latitude and longitude:

$$H_x = \begin{bmatrix} -\sin(lon) \\ \cos(lon) \\ 0 \end{bmatrix} \quad \text{East in ECEF}$$

$$H_y = \begin{bmatrix} -\sin(lat) * \cos(lon \\ -\sin(lat) * \sin(lon) \\ \cos(lat) \end{bmatrix} \quad \text{North in ECEF}$$

$$H_z = \begin{bmatrix} \cos(lat) * cos(lon) \\ \cos(lat) * \sin(lon) \\ \sin(lat) \end{bmatrix} \quad \text{Up in ECEF}$$

vi. Calculate the time of observation from the L1R line:
Time = Band Start Time + $L1R_L$ * Line Time
vii. Interpolate the spacecraft ECEF X, Y, Z position at the time of observation using 4 point Lagrange interpolation. This is implemented using IAS library functions, and yields the spacecraft ECEF vector $\mathbf{S_{ECEF}}$.
viii. Calculate the ground-to-space viewing vector:
$\mathbf{V_{ECEF}} = \mathbf{S_{ECEF}} - \mathbf{G_{ECEF}}$
ix. Project the ECEF viewing vector into the local vertical coordinate system:

$$\boldsymbol{V_{LV}} = \begin{bmatrix} H_x^T \\ H_y^T \\ H_z^T \end{bmatrix} \boldsymbol{V_{ECEF}}$$

This is equivalent to taking the dot product of the ECEF viewing vector with each of the local vertical system basis vectors.

x. Interpolate the ECEF sun direction vector at the time of observation using 4 point Lagrange interpolation. This is the same functionality used for the ephemeris data.

xi. Project the sun direction ECEF vector into the local vertical coordinate system as was done in step ix above, to yield the local vertical sun direction vector $\mathbf{S_{LV}}$.

b. As each grid point is processed, accumulate the sums of and then calculate the average values for the height, L1R line, L1R sample, L1T line, L1T sample, view vector X, Y, Z coordinates, and sun vector X, Y, Z coordinates.

c. For each component of the viewing vector $\mathbf{V_{LV}}$ and the sun vector $\mathbf{S_{LV}}$, compute the coefficients of a RPC model of the form shown above in equations 5a, 5b, and 5c.

i. Loop through all the vectors (each corresponding to a point in the grid) for this band to construct the normal equations:

1. The form of the observations is shown in equations (5a), (5b), and (5c) above. Each grid point yields one observation, expressed in matrix/vector notation: $X_{Satx}^T \theta_{Satx} = Y_{Satx}$

Where:

$$X_{Satx} = \begin{bmatrix} 1 \\ L1T_L \\ L1T_S \\ Hgt \\ L1R_L \\ L1T_L * L1T_L \\ L1T_L * L1T_S \\ L1T_S * L1T_S \\ L1R_S * L1R_L * L1R_L \\ L1R_L * L1R_L * L1R_L \\ -Sat_X * L1T_L \\ -Sat_X * L1T_S \\ -Sat_X * Hgt \\ -Sat_X * L1R_L \\ -Sat_X * L1T_L * L1T_L \\ -Sat_X * L1T_L * L1T_S \\ -Sat_X * L1T_S * L1T_S \\ -Sat_X * L1R_S * L1R_L * L1R_L \\ -Sat_X * L1R_L * L1R_L * L1R_L \end{bmatrix} \quad \theta_{Satx} = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \\ b_8 \\ b_9 \end{bmatrix}$$

$$Y_{Satx} = [Sat_X]$$

Note that all of the L1T, L1R, height, and vector component inputs above are offset by the means as shown in equation (5).

2. Each observation contributes to the normal equations:
   $N_{Satx}\theta_{Satx} = L_{Satx}$
   Where we have accumulated the N and L matrices as:
   $N_{Satx} \mathrel{+}= X_{Satx}X_{Satx}^T \qquad L_{Satx} \mathrel{+}= X_{Satx}Y_{Satx}$
   Once the observation contributions from all of the grid points are collected into the normal equation matrices, we solve for the unknown rational polynomial coefficient vectors θ_L and θ_S:
   $\theta_{Satx} = N_{Satx}^{-1}L_{Satx}$

d. This procedure is run six times, on the X, Y, and Z components of the view vector (Satx, Saty, and Satz) and on the X, Y, and Z components of the sun vector (Sunx, Suny, and Sunz). This generates a set of L1T-to-angle RPCs for the band.
e. Load the angle RPC model coefficients into the ANGCF structure.

8. Write the ANGCF structure to an output ODL formatted angle coefficient file.

a. Construct the output file name from the path, row, and date. This will need to be enhanced to include ground station ID and version number for operational use.
b. Write the FILE_HEADER group containing the file name, satellite ID, path, row, number of bands and band list.
c. Write the PROJECTION group containing the ellipsoid parameters, projection, units, datum, spheroid, and zone codes, the GCTP projection parameters, and the scene corner coordinates.
d. Write the EPHEMERIS group containing the ephemeris data start UTC epoch, the number of points, and the time from epoch, ECEF X, Y, and Z coordinates (in meters) for each point.
e. Write the SOLAR_VECTOR group containing the start UTC epoch, the number of points, and the time from epoch, ECEF X, Y, and Z directions for each point. The times will match the ephemeris data so those values are somewhat redundant.
f. Write an RPC_BANDnn group for each band containing the number of SCAs, SCA list, number of L1T lines and samples, number of L1R lines and samples, pixel size, band start UTC epoch and line time increment, mean height, mean L1T line/sample, mean L1R line/sample, mean view vector components, view vector RPC model coefficients, mean sun vector components, and mean sun vector RPC model coefficients.
    i. For each SCA in the band, write the mean height, mean L1T line/sample, mean L1R line/sample, and the L1T-to-L1R RPC model coefficients.

The output angle coefficient file contains all of the information needed by the phase 2 portion of the algorithm to generate satellite viewing and solar illumination angles for each L1T product pixel.

Phase 2: Compute Satellite Viewing and Solar Illumination Angles

The phase 2 experimental angle generation portion of the algorithm (simple_view) provides two options for performing the angle computations. Both methods use the L1T-to-L1R RPC models to calculate the L1R coordinates that correspond to a given L1T pixel. Both methods will also retrieve the pixel height from an input DEM, if provided. Otherwise, the mean elevation for the band, from the angle coefficient file, is used. Having been given the L1T line/sample and determined the corresponding L1R line/sample and height, the first, “rigorous”, method follows the procedure described above in step 7a of the phase 1 algorithm. The second, “RPC”, method applies equation (5) above using the parameters of the angle RPC model for the current band, stored in the angle coefficient file. The simpler operational prototype version (l8_angles) supports only the RPC and no-DEM options.

There are some subtleties to the use of the L1T-to-L1R RPCs that deserve some elaboration. Although the rational polynomial functions will generate L1R line and

sample coordinates for any given L1T product pixel (and height), each SCA has a separate set of RPCs, so it is necessary to know which SCA the pixel falls inside to select the correct model parameters. Having only the L1T image to work with, we will know which band number to use, but not which SCA. This was the original reason for including an L1R sample RPC model. We can evaluate the L1R sample coordinates for each SCA in the current band to decide which SCA, or SCAs, the L1T pixel came from. At most 2 SCAs will return L1R sample values that fall within the actual range of samples on that SCA, thus identifying the set of rational polynomial coefficients to use to evaluate the L1R line coordinate. In SCA overlap areas two SCAs will be valid. Depending upon the accuracy required, either could be used to compute the time and angles. In practice, both are evaluated and averaged, since overlapping pixels are averaged when the L1T products are generated.

Calculating Viewing and Solar Angles

The sequence of activities required to generate satellite viewing angles and solar illumination angles for each L1T pixel using the angle coefficients is as follows:

1. Capture the input command line parameters to determine which processing options to apply: RPC or rigorous computation, DEM input or mean height, subsampling factor.
2. Initialize the angle coefficient interface using the inputs provided:
    a. Open the input angle coefficient file and load the contents into an ANGCF data structure.
    b. Initialize the map projection logic:
        i. Construct a map projection structure using the parameters loaded in the ANGCF structure, by invoking the ias_geo_set_projection module.
        ii. Construct a geodetic projection structure to produce latitude/longitude coordinates in radians, using the ias_geo_set_projection module.
        iii. Construct a projection transformation that converts map X/Y to geodetic latitude/longitude using the structures from a. and b. above, and the ias_geo_create_proj_transformation module.
        iv. Pre-establishing this transformation will make subsequent map projection conversion computations easier.
    c. Load the GeoTIFF formatted DEM, if one is provided.
        i. Read the header information and store in a data structure.
        ii. Load the elevation array.
        iii. If no DEM is provided or the DEM does not match the image dimensions specified in the angle coefficient file, set the elevation array to NULL.
    d. Get the number of bands from the ANGCF structure and return this value to the calling procedure.
3. For each band:
    a. Extract the scene framing information from the ANGCF structure.

i. Extract the spectral band number, scene dimensions, map projection information (code, zone), pixel size, and upper-left corner coordinates from the ANGCF structure. The projection information will be used to generate the output angle image header files.

b. Calculate the size of the output angle images using the size of the L1T image and the subsampling factor:
Angle nlines = (L1T nlines – 1) / subsample + 1
Angle nsamps = (L1T nsamps – 1) / subsample + 1

c. Step through the L1T image pixels using the subsampling factor as a loop increment. Calculate the view and sun angles at each L1T line/sample location:
   i. Select the angle coefficients for the current band.
   ii. Calculate the subsampling relationship between the L1T image and the DEM:
   DEM subsample = (L1T nlines – 1) / (DEM nlines – 1)
   This is needed to properly index the DEM elevations when processing the panchromatic band.
   iii. Calculate the DEM indices that correspond to the current L1T indices by dividing by the DEM subsample factor, noting that the indices are zero-relative.
   iv. Extract the height from the DEM at the specified indices. If no DEM was provide, set the height to NULL.
   v. Calculate the angles using the selected method:
      1. Rigorous method – see below for details.
      2. RPC method – see below for details.
   vi. Quantize the computed angles to units of 0.01 degrees.

d. Write the angles to output band files:
   i. Calculate the angle band pixel size by multiplying the L1T pixel size by the subsampling factor.
   ii. Construct the output file names using the angle coefficient input file root name and the band number.
   iii. Write the satellite zenith and azimuth angle values, band sequentially, to the satellite angle file.
   iv. Write an ENVI-format header file for the satellite angles using the framing information extracted previously.
   v. Write the solar zenith and azimuth angles, band sequentially, to the solar angle file.
   vi. Write an ENVI-format header file for the solar angles using the framing information extracted previously.

4. Shut down the angle coefficient logic by releasing the allocated ephemeris data memory in the ANGCF structure and in the map projection transformation structure.

Computing Angles Using the Rigorous Method

To compute the satellite viewing and solar illumination angles at a specified L1T line/sample location, given the corresponding elevation and angle coefficients, using the rigorous method:

1. If the input height is NULL, replace it with the mean height from the band RPC parameters in the ANGCF structure.
2. Determine which SCA, or SCAs, viewed the L1T pixel:
    a. If the last_sca flag is invalid (e.g., for the first point calculated), start with the central SCA (isca = num_sca /2), otherwise use isca = last_sca.
    b. If the current SCA number (isca) is not valid (< 0 or >= num_SCA), return the number of valid SCAs found so far. Otherwise, compute the L1T-to-L1R RPCs for the current SCA:
        i. Offset the input L1T line, L1T sample, and height by the mean values for this band/SCA.
        ii. Evaluate L1R line and L1R sample using equations (1) and (2) above with the RPC coefficients for this band/SCA.
    c. If the computed L1R sample coordinate is between 0 and the number of L1R samples per SCA for this band.
        i. Increment the number of successful searches, ntry.
        ii. If the L1R line number is between 0 and the number of L1R lines in the image:
            1. Store the calculated L1R line coordinate.
            2. Convert the L1R sample SCA coordinate to a L1R file coordinate, and store that also:

            L1R_file_samp = L1R_SCA_samp + isca*num_samp_per_SCA

            3. Increment the number of SCAs found.
        iii. Set last_sca = isca
        iv. If we've found more than one SCA (ntry > 1) return the number found.
        v. If the L1R SCA sample number is below the SCA overlap threshold, decrement the current SCA index (isca) and go back to step b to test for a second overlapping SCA.
        vi. If the L1R SCA sample number is within the SCA overlap threshold of the number of samples per SCA, increment the current SCA index (isca) and go back to step b to test for a second overlapping SCA.
        vii. If the L1R SCA sample number is not within the potential overlap regions, return the number of SCAs found.
    d. If the L1R sample is out of range for the current SCA (i.e., the test in step c. above fails):
        i. If at least one SCA has already been found (ntry > 0) return the number found.
        ii. If the L1R sample number is outside the image (< 0 for the first SCA or > number of samples for the last SCA), return the number of SCAs found.

iii. Calculate the L1R file sample number:
L1R_file_samp = L1R_SCA_samp + isca * num_samp_per_SCA
iv. Calculate the predicted SCA index:
isca = L1R_file_samp / num_samp_per_SCA
isca = MAX(isca, 0)
isca = MIN(isca, num_SCA – 1)
v. Go back to step b.

e. This sub-algorithm returns the number of valid SCAs found and the corresponding L1R line and sample coordinates for each.

3. For each SCA found to contain the point, calculate the satellite and solar vectors:
   a. This procedure is described in step 7.a. of the phase 1 algorithm above with the exception of the first, height retrieval, sub-step. Here, the height is provided as an input.
4. Calculate the satellite and sun zenith angles corresponding to the vectors, clipping the zenith angles at the horizon (90 degrees):

```
if  satvector.z > 0 then sat_zenith = acos( satvector.z )
else sat_zenith = 0
if sunvector.z > 0 then sun_zenith = acos( sunvector.z )
else sun_zenith = 0
```

5. Calculate the satellite and sun azimuth angles, setting the azimuth equal to zero if the vector is vertical:

```
hdist = sqrt( satvector.x*satvector.x + satvector.y*satvector.y )
if hdist > 0 then sat_azimuth = atan2( satvector.x, satvector.y )
else sat_azimuth = 0
hdist = sqrt( sunvector.x*sunvector.x + sunvector.y*sunvector.y )
if hdist > 0 then sun_azimuth = atan2( sunvector.x, sunvector.y )
else sun_azimuth = 0
```

6. Average the angles computed from the individual SCAs.

Computing Angles Using the Rational Polynomial Coefficient (RPC) Method

To compute the satellite viewing and solar illumination angles at a specified L1T line/sample location, given the corresponding elevation and angle coefficients, using the RPC method:

1. If the input height is NULL, replace it with the mean height from the band RPC parameters in the ANGCF structure.
2. Determine which SCA, or SCAs, viewed the L1T pixel. This procedure is the same as for the rigorous method and is described above. Note, however, that this sub-algorithm returns the L1R line and L1R file sample coordinates for each valid SCA. The L1R sample coordinate was not used in the rigorous method, but will be here.
3. Offset the L1T line, L1T sample, and height values by the mean values for the current band.
4. For each valid SCA:

a. Offset the L1R line and L1R sample coordinates by the mean values for the current band.
b. Use the offset values and the angle RPC model parameters for this band to evaluate equations (5a), (5b), and (5c) above for each component of the satellite viewing vector and each component of the solar illumination vector.
c. Calculate the angles corresponding to the resulting vectors using the methods described in steps 4 and 5 of the rigorous method, above.

5. Average the angles computed from the individual SCAs.

Angle Coefficient Output File

The detailed contents of the angle coefficient (ANG) file are shown in Table 4-6 below. Note that although some of the fields in the ANG file duplicate information found in the standard L1T product metadata (MTA) file, this was done intentionally to make the ANG file self- contained. In some cases, different parameter names are used in the ANG file. In a production implementation it may be desirable to harmonize the parameter names or even combine the files into one. Such decisions are beyond the scope of this algorithm.

The ANG file is ODL structure text and consists of 15 parameter groups: a file header group, a projection group, an ephemeris group, a solar vector group, and one group of RPC model parameters for each of the 11 spectral bands.

Group	Parameter	Type	Size	Contents
FILE_HEADER	FILE_NAME	char	29	The ANG file name mimics the MTA file name with the extension “MTA” replaced by “ANG”.
FILE_HEADER	SATELLITE	char	9	Satellite identifier = LANDSAT_9
FILE_HEADER	WRS_PATH	int	1	Scene WRS-2 orbit-based path (1-233)
FILE_HEADER	WRS_ROW	int	1	Scene WRS-2 orbit-based row (1-248)
FILE_HEADER	NUMBER_OF_BANDS	int	1	Number of bands contained in this file, normally 11.
FILE_HEADER	BAND_LIST	int	11	List of the spectral band numbers contained in this file, normally (1,2,3,4,5,6,7,8,9,10,11).
PROJECTION	ELLIPSOID_AXES	double	2	WGS84 ellipsoid semi-major and semi-minor axes in meters.
PROJECTION	PROJECTION_CODE	int	1	Code for map projection type: 1 = UTM, 6 = PS.
PROJECTION	PROJECTION_UNITS	char	6	Map projection units will always be METERS
PROJECTION	PROJECTION_DATUM	char	5	Datum will always be WGS84
PROJECTION	PROJECTION_SPHEROID	int	1	The projection spheroid code will always be 12.
PROJECTION	PROJECTION_ZONE	int	1	UTM zone number (1-60). Note that only northern hemisphere zones are used so this number will always be positive.
PROJECTION	PROJECTION_PARAMETERS	double	15	GCTP map projection parameters. All zeros for UTM. For polar stereographic this contains the ellipsoid axes, false easting and northing (both 0), latitude of true scale (+/-71 degrees) and the vertical axis longitude (0).
PROJECTION	UL_CORNER	double	2	L1T upper left corner map projection coordinates (meters).
PROJECTION	UR_CORNER	double	2	L1T upper right corner map projection coordinates (meters).
PROJECTION	LL_CORNER	double	2	L1T lower left corner map projection coordinates (meters).
PROJECTION	LR_CORNER	double	2	L1T lower right corner map projection coordinates (meters).
EPHEMERIS	EPHEMERIS_EPOCH_YEAR	int	1	Year of ephemeris epoch (start time).
EPHEMERIS	EPHEMERIS_EPOCH_DAY	int	1	Epoch day of year.
EPHEMERIS	EPHEMERIS_EPOCH_SECOND	double	1	Epoch seconds of day.
EPHEMERIS	NUMBER_OF_POINTS	int	1	Number of ephemeris points provided in following four parameter fields.
EPHEMERIS	EPHEMERIS_TIME	double	variable	Ephemeris sample time offsets (from epoch) in seconds.
EPHEMERIS	EPHEMERIS_ECEF_X	double	variable	Ephemeris sample Earth Centered Earth Fixed X coordinate in meters.

Group	Parameter	Type	Size	Contents
EPHEMERIS	EPHEMERIS_ECEF_Y	double	variable	Ephemeris sample Earth Centered Earth Fixed Y coordinate in meters.
EPHEMERIS	EPHEMERIS_ECEF_Z	double	variable	Ephemeris sample Earth Centered Earth Fixed Z coordinate in meters.
SOLAR_VECTOR	SOLAR_EPOCH_YEAR	int	1	Year of solar vector epoch (start time). This is the same as the ephemeris epoch.
SOLAR_VECTOR	SOLAR_EPOCH_DAY	int	1	Epoch day of year.
SOLAR_VECTOR	SOLAR_EPOCH_SECOND	double	1	Epoch seconds of day.
SOLAR_VECTOR	NUMBER_OF_POINTS	int	1	Number of solar vectors provided in following four parameter fields.
SOLAR_VECTOR	SAMPLE_TIME	double	variable	Vector sample time offsets (from epoch) in seconds.
SOLAR_VECTOR	SOLAR_ECEF_X	double	variable	Solar vector sample Earth Centered Earth Fixed X direction.
SOLAR_VECTOR	SOLAR_ECEF_Y	double	variable	Solar vector sample Earth Centered Earth Fixed Y direction.
SOLAR_VECTOR	SOLAR_ECEF_Z	double	variable	Solar vector sample Earth Centered Earth Fixed Z direction.
RPC_BAND01	NUMBER_OF_SCAS	int	1	Number of SCAs: 14 for OLI bands.
RPC_BAND01	NUM_L1T_LINES	int	1	Number of lines in the L1T product.
RPC_BAND01	NUM_L1T_SAMPS	int	1	Number of samples in the L1T product.
RPC_BAND01	NUM_L1R_LINES	int	1	Number of lines in the L1R product.
RPC_BAND01	NUM_L1R_SAMPS	int	1	Number of samples per SCA in the L1R product.
RPC_BAND01	PIXEL_SIZE	double	1	L1T pixel size, in meters.
RPC_BAND01	START_TIME	double	1	L1R image start time in seconds from the ephemeris epoch.
RPC_BAND01	LINE_TIME	double	1	L1R image line time increment in seconds.
RPC_BAND01	BAND01_MEAN_HEIGHT	double	1	Mean height offset for the RPC angle model.
RPC_BAND01	BAND01_MEAN_L1R_LINE_SAMP	double	2	Mean L1R line and (file) sample offsets for the RPC angle model.
RPC_BAND01	BAND01_MEAN_L1T_LINE_SAMP	double	2	Mean L1T line and sample offsets for the RPC angle model.
RPC_BAND01	BAND01_MEAN_SAT_VECTOR	double	3	Mean satellite view vector for the RPC angle model.
RPC_BAND01	BAND01_SAT_X_NUM_COEF	double	10	Numerator polynomial coefficients for the satellite view vector X coordinate.
RPC_BAND01	BAND01_SAT_X_DEN_COEF	double	9	Denominator polynomial coefficients for the satellite view vector X coordinate.
RPC_BAND01	BAND01_SAT_Y_NUM_COEF	double	10	Numerator polynomial coefficients for the satellite view vector Y coordinate.

Group	Parameter	Type	Size	Contents
RPC_BAND01	BAND01_SAT_Y_DEN_COEF	double	9	Denominator polynomial coefficients for the satellite view vector Y coordinate.
RPC_BAND01	BAND01_SAT_Z_NUM_COEF	double	10	Numerator polynomial coefficients for the satellite view vector Z coordinate.
RPC_BAND01	BAND01_SAT_Z_DEN_COEF	double	9	Denominator polynomial coefficients for the satellite view vector Z coordinate.
RPC_BAND01	BAND01_MEAN_SUN_VECTOR	double	3	Mean sun vector for the RPC angle model.
RPC_BAND01	BAND01_SUN_X_NUM_COEF	double	10	Numerator polynomial coefficients for the sun vector X coordinate.
RPC_BAND01	BAND01_SUN_X_DEN_COEF	double	9	Denominator polynomial coefficients for the sun vector X coordinate.
RPC_BAND01	BAND01_SUN_Y_NUM_COEF	double	10	Numerator polynomial coefficients for the sun vector Y coordinate.
RPC_BAND01	BAND01_SUN_Y_DEN_COEF	double	9	Denominator polynomial coefficients for the sun vector Y coordinate.
RPC_BAND01	BAND01_SUN_Z_NUM_COEF	double	10	Numerator polynomial coefficients for the sun vector Z coordinate.
RPC_BAND01	BAND01_SUN_Z_DEN_COEF	double	9	Denominator polynomial coefficients for the sun vector Z coordinate.
RPC_BAND01	BAND01_SCA_LIST	int	14	List of SCAs in this band. For OLI bands this is: (1,2,3,4,5,6,7,8,9,10,11,12,13,14)
RPC_BAND01	BAND01_SCA01_MEAN_HEIGHT	double	1	Mean height offset for the SCA01 L1T-to-L1R RPC model.
RPC_BAND01	BAND01_SCA01_MEAN_L1R_LINE_SAMP	double	2	Mean L1R line and (SCA) sample offsets for the SCA01 L1T-to-L1R RPC model.
RPC_BAND01	BAND01_SCA01_MEAN_L1T_LINE_SAMP	double	2	Mean L1T line and sample offsets for the SCA01 L1T-to-L1R RPC model.
RPC_BAND01	BAND01_SCA01_LINE_NUM_COEF	double	5	Numerator polynomial coefficients for the SCA01 L1R line RPC model.
RPC_BAND01	BAND01_SCA01_LINE_DEN_COEF	double	4	Denominator polynomial coefficients for the SCA01 L1R line RPC model.
RPC_BAND01	BAND01_SCA01_SAMP_NUM_COEF	double	5	Numerator polynomial coefficients for the SCA01 L1R sample RPC model.
RPC_BAND01	BAND01_SCA01_SAMP_DEN_COEF	double	4	Denominator polynomial coefficients for the SCA01 L1R sample RPC model.
RPC_BAND01	BAND01_SCA02_MEAN_HEIGHT	double	1	Mean height offset for the SCA02 L1T-to-L1R RPC model.

Group	Parameter	Type	Size	Contents
RPC_BAND01	BAND01_SCA02_MEAN_L1R_LINE_SAMP	double	2	Mean L1R line and (SCA) sample offsets for the SCA02 L1T-to-L1R RPC model.
RPC_BAND01	BAND01_SCA02_MEAN_L1T_LINE_SAMP	double	2	Mean L1T line and sample offsets for the SCA02 L1T-to-L1R RPC model.
RPC_BAND01	BAND01_SCA02_LINE_NUM_COEF	double	5	Numerator polynomial coefficients for the SCA02 L1R line RPC model.
RPC_BAND01	BAND01_SCA02_LINE_DEN_COEF	double	4	Denominator polynomial coefficients for the SCA02 L1R line RPC model.
RPC_BAND01	BAND01_SCA02_SAMP_NUM_COEF	double	5	Numerator polynomial coefficients for the SCA02 L1R sample RPC model.
RPC_BAND01	BAND01_SCA02_SAMP_DEN_COEF	double	4	Denominator polynomial coefficients for the SCA02 L1R sample RPC model.
RPC_BAND01	BAND01_SCAnn_...			The previous seven parameters repeat for SCAs 03 through 14.
…				
RPC_BANDmm				The RPC_BAND01 group parameters are repeated for bands 2 through 9.
…				
RPC_BAND10	NUMBER_OF_SCAS	int	1	Number of SCAs: 3 for TIRS bands.
RPC_BAND10	NUM_L1T_LINES	int	1	Number of lines in the L1T product.
RPC_BAND10	NUM_L1T_SAMPS	int	1	Number of samples in the L1T product.
RPC_BAND10	NUM_L1R_LINES	int	1	Number of lines in the L1R product.
RPC_BAND10	NUM_L1R_SAMPS	int	1	Number of samples per SCA in the L1R product.
RPC_BAND10	PIXEL_SIZE	double	1	L1T pixel size, in meters.
RPC_BAND10	START_TIME	double	1	L1R image start time in seconds from the ephemeris epoch.
RPC_BAND10	LINE_TIME	double	1	L1R image line time increment in seconds.
RPC_BAND10	BAND10_MEAN_HEIGHT	double	1	Mean height offset for the RPC angle model.
RPC_BAND10	BAND10_MEAN_L1R_LINE_SAMP	double	2	Mean L1R line and (file) sample offsets for the RPC angle model.
RPC_BAND10	BAND10_MEAN_L1T_LINE_SAMP	double	2	Mean L1T line and sample offsets for the RPC angle model.
RPC_BAND10	BAND10_MEAN_SAT_VECTOR	double	3	Mean satellite view vector for the RPC angle model.
RPC_BAND10	BAND10_SAT_X_NUM_COEF	double	10	Numerator polynomial coefficients for the satellite view vector X coordinate.
RPC_BAND10	BAND10_SAT_X_DEN_COEF	double	9	Denominator polynomial coefficients for the satellite view vector X coordinate.

Group	Parameter	Type	Size	Contents
RPC_BAND10	BAND10_SAT_Y_NUM_COEF	double	10	Numerator polynomial coefficients for the satellite view vector Y coordinate.
RPC_BAND10	BAND10_SAT_Y_DEN_COEF	double	9	Denominator polynomial coefficients for the satellite view vector Y coordinate.
RPC_BAND10	BAND10_SAT_Z_NUM_COEF	double	10	Numerator polynomial coefficients for the satellite view vector Z coordinate.
RPC_BAND10	BAND10_SAT_Z_DEN_COEF	double	9	Denominator polynomial coefficients for the satellite view vector Z coordinate.
RPC_BAND10	BAND10_MEAN_SUN_VECTOR	double	3	Mean sun vector for the RPC angle model.
RPC_BAND10	BAND10_SUN_X_NUM_COEF	double	10	Numerator polynomial coefficients for the sun vector X coordinate.
RPC_BAND10	BAND10_SUN_X_DEN_COEF	double	9	Denominator polynomial coefficients for the sun vector X coordinate.
RPC_BAND10	BAND10_SUN_Y_NUM_COEF	double	10	Numerator polynomial coefficients for the sun vector Y coordinate.
RPC_BAND10	BAND10_SUN_Y_DEN_COEF	double	9	Denominator polynomial coefficients for the sun vector Y coordinate.
RPC_BAND10	BAND10_SUN_Z_NUM_COEF	double	10	Numerator polynomial coefficients for the sun vector Z coordinate.
RPC_BAND10	BAND10_SUN_Z_DEN_COEF	double	9	Denominator polynomial coefficients for the sun vector Z coordinate.
RPC_BAND10	BAND10_SCA_LIST	int	3	List of SCAs in this band. For TIRS bands this is: (1,2,3)
RPC_BAND10	BAND10_SCA01_MEAN_HEIGHT	double	1	Mean height offset for the SCA01 L1T-to-L1R RPC model.
RPC_BAND10	BAND10_SCA01_MEAN_L1R_LINE_SAMP	double	2	Mean L1R line and (SCA) sample offsets for the SCA01 L1T-to-L1R RPC model.
RPC_BAND10	BAND10_SCA01_MEAN_L1T_LINE_SAMP	double	2	Mean L1T line and sample offsets for the SCA01 L1T-to-L1R RPC model.
RPC_BAND10	BAND10_SCA01_LINE_NUM_COEF	double	5	Numerator polynomial coefficients for the SCA01 L1R line RPC model.
RPC_BAND10	BAND10_SCA01_LINE_DEN_COEF	double	4	Denominator polynomial coefficients for the SCA01 L1R line RPC model.
RPC_BAND10	BAND10_SCA01_SAMP_NUM_COEF	double	5	Numerator polynomial coefficients for the SCA01 L1R sample RPC model.
RPC_BAND10	BAND10_SCA01_SAMP_DEN_COEF	double	4	Denominator polynomial coefficients for the SCA01 L1R sample RPC model.
RPC_BAND10	BAND10_SCA02_MEAN_HEIGHT	double	1	Mean height offset for the SCA02 L1T-to-L1R RPC model.

Group	Parameter	Type	Size	Contents
RPC_BAND10	BAND10_SCA02_MEAN_L1R_LINE_SAMP	double	2	Mean L1R line and (SCA) sample offsets for the SCA02 L1T-to-L1R RPC model.
RPC_BAND10	BAND10_SCA02_MEAN_L1T_LINE_SAMP	double	2	Mean L1T line and sample offsets for the SCA02 L1T-to-L1R RPC model.
RPC_BAND10	BAND10_SCA02_LINE_NUM_COEF	double	5	Numerator polynomial coefficients for the SCA02 L1R line RPC model.
RPC_BAND10	BAND10_SCA02_LINE_DEN_COEF	double	4	Denominator polynomial coefficients for the SCA02 L1R line RPC model.
RPC_BAND10	BAND10_SCA02_SAMP_NUM_COEF	double	5	Numerator polynomial coefficients for the SCA02 L1R sample RPC model.
RPC_BAND10	BAND10_SCA02_SAMP_DEN_COEF	double	4	Denominator polynomial coefficients for the SCA02 L1R sample RPC model.
RPC_BAND10	BAND10_SCA03_MEAN_HEIGHT	double	1	Mean height offset for the SCA03 L1T-to-L1R RPC model.
RPC_BAND10	BAND10_SCA03_MEAN_L1R_LINE_SAMP	double	2	Mean L1R line and (SCA) sample offsets for the SCA03 L1T-to-L1R RPC model.
RPC_BAND10	BAND10_SCA03_MEAN_L1T_LINE_SAMP	double	2	Mean L1T line and sample offsets for the SCA03 L1T-to-L1R RPC model.
RPC_BAND10	BAND10_SCA03_LINE_NUM_COEF	double	5	Numerator polynomial coefficients for the SCA03 L1R line RPC model.
RPC_BAND10	BAND10_SCA03_LINE_DEN_COEF	double	4	Denominator polynomial coefficients for the SCA03 L1R line RPC model.
RPC_BAND10	BAND10_SCA03_SAMP_NUM_COEF	double	5	Numerator polynomial coefficients for the SCA03 L1R sample RPC model.
RPC_BAND10	BAND10_SCA03_SAMP_DEN_COEF	double	4	Denominator polynomial coefficients for the SCA03 L1R sample RPC model.
RPC_BAND11				The RPC_BAND10 group parameters are repeated for band 11.

Table 4-6. Angle Coefficient File Detailed Content

4.2 OLI Geometry Algorithms

4.2.1 OLI Line-of-Sight Model Creation Algorithm

4.2.1.1 Background/Introduction

The LOS model creation algorithm gathers the ancillary data and calibration parameters required to support geometric processing of the input image dataset, validates the image time codes, extracts the corresponding ephemeris and attitude data from the ancillary data stream, performs the necessary coordinate transformations, and stores the results in a geometric model structure for subsequent use by other geometric algorithms. The OLI LOS model creation algorithm was originally derived from the ALI model creation algorithm. Much of the ephemeris and attitude preprocessing logic present in the ALI implementation has been moved to the (now separate) ancillary data preprocessing algorithm to better isolate the bulk of the geometric processing logic from the details of the incoming ancillary data stream. New attitude data processing logic has also been added to separate the high- and low-frequency attitude effects to allow the image resampling process to better correct for jitter at frequencies above the original 10 Hz algorithm design limit, without requiring an unreasonably dense resampling grid.

4.2.1.2 Dependencies

The LOS Model Creation algorithm assumes that the Ancillary Data Preprocessing algorithm has been executed to accomplish the following:

- Validated ephemeris data for the full imaging interval have been generated
- Validated attitude data for the full imaging interval have been generated
- The ancillary data have been scaled to engineering units
- The Ingest process has applied the spacecraft clock epoch offset from J2000 to convert all spacecraft times to seconds from J2000. This offset is zero for L8 but is 0.184 seconds for L9.

Whether or not “definitive” processing has been performed, the Ancillary Data Preprocessing algorithm will generate preprocessed smoothed and cleaned ephemeris and attitude data streams. The format will be the same for either validated spacecraft estimates or definitive processing. No definitive ephemeris or attitude processing was required for Landsat 8 and none is anticipated for Landsat 9.

4.2.1.3 Inputs

The LOS Model Creation algorithm uses the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data; including dataset IDs to provide unique identifiers for data trending).

Algorithm Inputs
ODL File (implementation)
Acquisition Type (Earth, Lunar, Stellar) (optional, defaults to Earth)
CPF File Name
Ancillary Data Input File Name

Algorithm Inputs
L0R/L1R Directory and File Names
WRS Path/Row (stored in model and used for trending)
Trending On/Off Switch (not implemented in prototype)
L0Rp ID (for trending)
Work Order ID (for trending)
Optional Precision Model Input Parameters (see note 9)
Input Precision Model Reference Time (optional)
Input Precision Ephemeris Correction Order (optional)
Input Precision X Correction Parameters (optional)
Input Precision Y Correction Parameters (optional)
Input Precision Z Correction Parameters (optional)
Input Precision Attitude Correction Order (optional)
Input Precision Roll Correction Parameters (optional)
Input Precision Pitch Correction Parameters (optional)
Input Precision Yaw Correction Parameters (optional)
CPF Contents
WGS84 Earth ellipsoid parameters
Earth orientation parameters (UT1UTC, pole wander, leap seconds) (see note 1)
Earth rotation velocity
Speed of light
ACS to OLI rotation matrix
Spacecraft center of mass (CM) to OLI offset in ACS reference frame (meters)
High-frequency attitude data cutoff frequency (Hz)
Focal plane model parameters (Legendre coefficients)
Detector delay table (including whole pixel even/odd and deselect offsets)
Nominal L0R fill (per band)
Nominal OLI frame time nominal_frame_time (4.236 msec)
Nominal OLI MS and pan integration times (msec)
OLI MS and pan detector settling times (msec)
Preprocessed Ancillary Data Contents
Attitude Data
Attitude data UTC epoch: Year, Day of Year, Seconds of Day
Time from epoch (one per sample, nominally 50 Hz)
ECI quaternion (vector: q1, q2, q3, scalar: q4) (one per sample)
ECEF quaternion (one per sample)
Body rate estimate (roll, pitch, yaw rate) (one per sample)
Roll, pitch, yaw estimate (one per sample)
Ephemeris Data
Ephemeris data UTC epoch: Year, Day of Year, Seconds of Day
Time from epoch (one per sample, nominally 1 Hz)
ECI position estimate (X, Y, Z) (one set per sample)
ECI velocity estimate (Vx, Vy, Vz) (one set per sample)
ECEF position estimate (X, Y, Z) (one set per sample)
ECEF velocity estimate (Vx, Vy, Vz) (one set per sample)
L0R/L1R Data Contents
Image Time Codes (one per line)
Integration Time (one value for MS bands and one value for pan band)
Detector Alignment Fill Table (see note 2)

4.2.1.4 Outputs

OLI LOS Model (see Table 4-7 below)
WGS84 Earth ellipsoid parameters
Earth Orientation Parameters (for current day) from CPF
Earth rotation velocity
Speed of light
OLI to ACS reference alignment matrix/quaternion
Spacecraft center of mass to OLI offset in ACS reference frame
Focal plane model parameters (Legendre coefs)
Detector delay table (now including whole pixel even/odd and deselect offsets)
Nominal detector alignment fill table (from CPF)
L0R detector alignment fill table (from L0R)
ECI J2000 spacecraft ephemeris model (original and corrected)
ECEF spacecraft ephemeris model (original and corrected)
Spacecraft attitude model (time, roll, pitch, yaw) (orig and corr) (see note 4)
High frequency attitude perturbations (roll, pitch, yaw) per image line (jitter table)
Image time codes (see note 5) (in seconds)
Integration Time (MS and pan) (in seconds)
Sample Time (MS and pan) (in seconds)
Settling Time (MS and pan) (in seconds)
WRS Path/Row
Model Trending Data
WRS Path/Row
L0Rp ID
Work Order ID
Image start UTC time (year, day of year, seconds of day)
Computed image frame time (in seconds)
Number of image lines
Number of out-of-limit image time codes

4.2.1.5 Options

Trending On/Off Switch
Optional precision model input parameters can be used to force model corrections.

4.2.1.6 Procedure

The LOS model is stored as a structure and is created from information contained in the Level 0R or Level 1R image data, the CPF, and the Ancillary data. The model is subsequently used, along with the CPF, to create a resampling grid. Data present in the model structure includes satellite position, velocity, and attitude, LOS angles, timing references, precision-correction information (if any), and the software version. The LOS model is also used in several characterization and calibration routines for mapping input line/sample locations to geographic latitude/longitude.

The LOS model may be thought of in two parts, an instrument model that provides a line-of-sight vector for each OLI detector (and, hence, each image line/sample), and a spacecraft model that provides spacecraft ephemeris (position and velocity) and attitude

as a function of time. These models are linked by the image time stamps that allow each Level 0R or Level 1R image sample to be associated with a time of observation.

4.2.1.6.1 Instrument Model

The model treats every band of every SCA independently. This is done by defining a set of 2nd order Legendre polynomials for each band of each SCA. Since the odd and even detectors are staggered for each band (Figure 4-19) as well as there being multiple pixel selects, the set of Legendre polynomials represent a theoretical "nominal" set of detectors that are best-fit to the even detectors for the first pixel select. This approach treats the odd detectors and second and third pixel select detectors as though they were aligned with the even detectors for the first pixel select for purposes of sensor LOS generation. This approach also explicitly models the slight offsets caused by the actual odd detector offset, any offsets caused by detector deselect, and the subpixel deviations of each detector from its nominal location, for correction during image resampling. This leads to four detector types: nominal, actual, maximum, and exact. A nominal detector is calculated from the Legendre polynomials. An actual detector corrects the nominal detector location for the whole pixel odd/even and pixel select offsets. For the ALI, these offsets were band dependent. For the OLI, since individual detectors may be deselected, they are detector dependent. The maximum detector option uses the largest possible even/odd and pixel select offset for a given band. This is used to compute detector terrain parallax sensitivity coefficients when generating the line-of-sight grid. See the LOS Projection/Grid Generation Algorithm description in Section 4.2.2 for additional details. An exact detector has the actual correction applied but also includes the specific individual (subpixel) detector offsets. The Legendre polynomials and a table of detector-offset values are stored in the CPF.

There is a slight angular difference between the line-of-sight vectors or angles associated with the odd/even and multiple pixel select detectors. If the nominal LOS, generated using the 2nd order Legendre model, is $\psi_{nominal}$, the look angles for the actual and exact detectors are as follows:

$$\psi_{x\_actual} = \psi_{x\_nominal} + \text{round}(\text{detector\_shift\_x}) * \text{IFOV}$$
$$\psi_{y\_actual} = \psi_{y\_nominal} + \text{round}(\text{detector\_shift\_y}) * \text{IFOV}$$

$$\psi_{x\_exact} = \psi_{x\_nominal} + \text{detector\_shift\_x} * \text{IFOV}$$
$$\psi_{y\_exact} = \psi_{y\_nominal} + \text{detector\_shift\_y} * \text{IFOV}$$

The maximum detector case uses the largest possible along-track offset and the nominal across-track offset, which is zero:

$$\psi_{x\_maximum} = \psi_{x\_nominal} + \text{maximum\_shift\_x} * \text{IFOV}$$
$$\psi_{y\_maximum} = \psi_{y\_nominal}$$

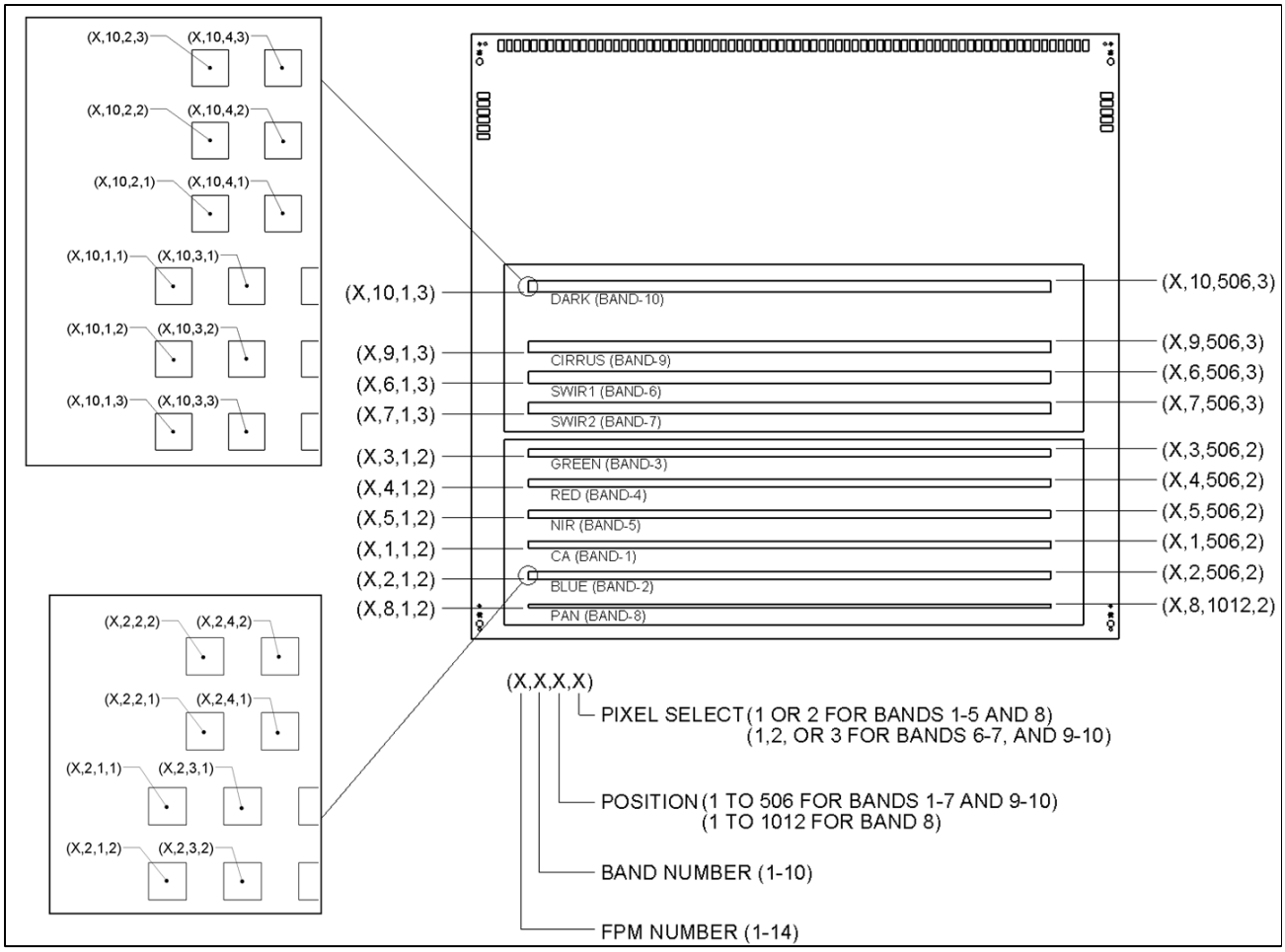

Figure 4-19. Detector Layout

The detector_shift_x and detector_shift_y values are the detector-specific offsets stored in the CPF detector delay tables. These offsets include both the whole-pixel even/odd and deselect offsets and the fractional-pixel detector placement effects, and must be rounded to extract the integer portion. Note that the integer portion of the detector_shift_y value is always zero since the even/odd and deselect effects are applicable only in the X direction.

The nominal LOS is used in most line-of-sight projection applications. The actual LOS is used in conjunction with the actual image time (see below) to model the errors introduced by trading time (sample delay) for space (detector offset) for purposes of correcting the nominal LOS model. The exact LOS is generally used only for data simulation and other analytical purposes rather than in the geometric correction model, as the subpixel portion of the detector delay is applied directly in the image resampler rather than being included in the LOS model.

4.2.1.6.1.1 Sample Timing

The OLI provides a time stamp with each image frame collected. These time stamps make it possible to relate the image samples (pixels) to the corresponding spacecraft

ephemeris and attitude data. Figure 4-20 shows the OLI sample timing relationships. Several items in this figure are worthy of particular note. First, the time stamp associated with a data frame is recorded at the end of the detector integration time. Second, there is a small settling and sampling delay (MS SS and Pan SS in the figure) between the end of detector integration and time stamp generation. Third, the time stamps are delayed by one data frame so that data frame N contains the time stamp for data frame N-1. Fourth, the data frame associated with time stamp N contains the multispectral (MS) samples collected just prior to time stamp N as well as the panchromatic samples collected just prior to and just after time stamp N, rather than the two samples collected prior to time stamp N. This is important for relating the panchromatic sample timing to the multispectral sample timing.

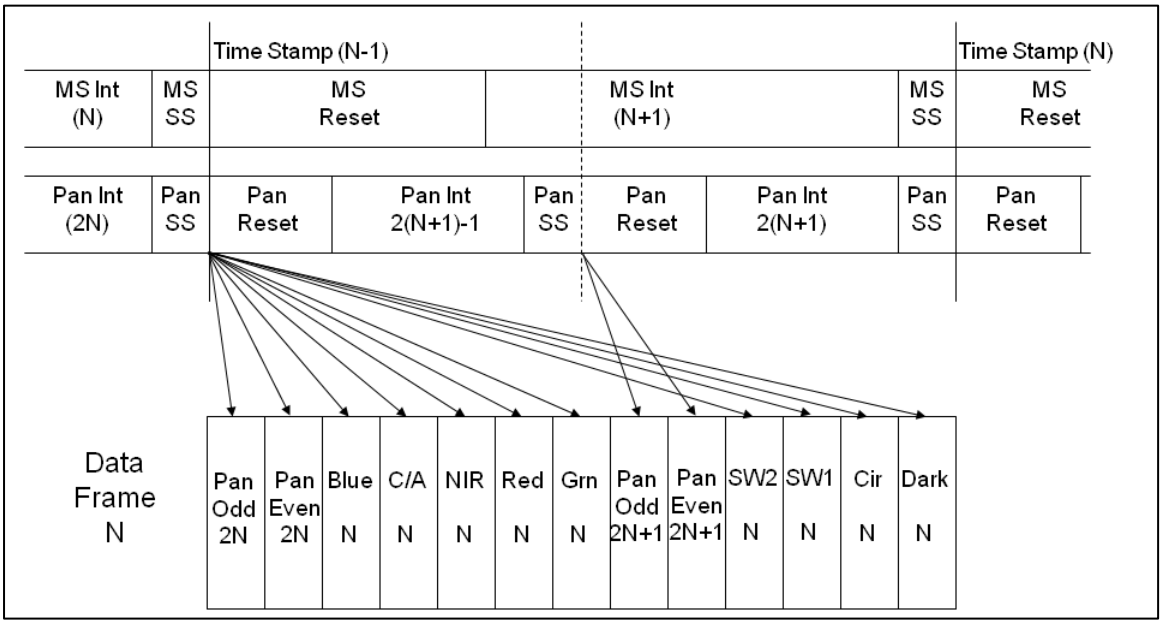

Figure 4-20. OLI Focal Plane Electronics Detector Sample Timing Diagram

One further complication to the problem of assigning times to image samples is the fact that the Level 0R/1R input imagery may include fill pixels inserted to achieve nominal even/odd detector alignment. This fill insertion allows the geometrically unprocessed 0R/1R imagery to be viewed as a spatially contiguous image without even/odd detector misalignments. The amount of detector alignment fill present will be indicated in the L0R/L1R image data (this is the purpose of the detector alignment fill table input noted above) so that the association of image samples with their corresponding time stamps can be adjusted accordingly. In the heritage ALIAS system, fill pixels were also inserted to achieve nominal band alignment. This will not be done for OLI data.

Due to the staggered odd/even and multiple pixel select detectors, a nominal and an actual time can be found in a scene. The actual time reflects the time that the current detector was actually sampled, whereas the nominal time reflects the time at which the idealized detector represented by the OLI LOS model would have been sampled. There

is also a “maximum” detector time option used in the computation of detector terrain parallax sensitivity coefficients during grid generation.

If the current position within the image is given as a line and sample location, the different types of times for multispectral pixels are calculated by the following:

```
if detector_type is set to MAXIMUM
    l0r_fill_pixels = nominal_fill_pixels + round(maximum_detector_delay)
else
    l0r_fill_pixels = Fill value for current detector from L0Rp

time_index = round( MS_line ) - l0r_fill_pixels + 1
if ( time_index < 0 ) time_index = 0
if (time_index > (num_time_stamps - 1)) time_index = num_time_stamps - 1

MS_actual_time = line_time_stamp[time_index] - MS_settle_time
                 - MS_integration_time/2
                 + (MS_line - l0r_fill_pixels – (time_index-1)) * MS_sample_time

MS_nominal_time = MS_actual_time
                  + (l0r_fill_pixels – nominal_fill_pixels) * MS_sample_time
```

where:

- MS_line is the zero-referenced multispectral line number (N).
- nominal_fill_pixels is the amount of even/odd detector alignment fill to be inserted at the beginning of pixel columns that correspond to nominal detectors; that is, those detectors with a delay value of zero that are the basis for the Legendre polynomial LOS model. This value comes from the CPF.
- l0r_fill_pixels is the total amount of even/odd detector alignment fill inserted at the beginning of the pixel column associated with the current detector in the Level 0R image data. It includes both the nominal_fill_pixels and the detector-specific delay fill required to align even/odd detectors.
- num_time_stamps is the total number of time codes (data frames) in the image. It is tested to ensure that time_index, the line_time_stamp index, does not go out of bounds.
- maximum_detector_delay is a constant offset that represents the largest amount of even/odd detector offset for any detector from the LOS model detector delay table. It is rounded to the nearest integer pixel because time offsets can only occur in whole line increments. The value of this parameter is not critical, as the line-of-sight offsets computed for “maximum” detectors are divided by the maximum delay to compute offset-per-unit-delay coefficients. This parameter is set in a #define statement.
- MS_settle_time is a small sample and hold time delay constant.

The MS_settle_time correction is a small (tens of microseconds) constant offset that is captured in the CPF. The L0R/L1R data can be accessed by SCA making the association of sample number with detector index more straightforward. Note that the Level 0R data inverts the detector read-out order for the even-numbered SCAs so that the samples are numbered left-to-right for all SCAs. This convention is also followed in the CPF detector offset tables. Also note that the non-imaging detector data (video reference pixels) are stored separately from the image data in the L0R and are not modeled in the CPF. Therefore, there are 494 samples per SCA in the multispectral bands and 988 samples per SCA in the panchromatic band.

For the panchromatic band, the corresponding equations for a pan detector in the two pan lines (2N and 2N+1) associated with MS line N (Figure 4-20) are computed as follows:

```
if detector_type is set to MAXIMUM
    l0r_fill_pixels = nominal_fill_pixels + round(maximum_detector_delay)
else
    l0r_fill_pixels = Fill value for current detector from L0Rp

time_index = floor( (round( pan_line ) - l0r_fill_pixels)/2 ) + 1
if ( time_index < 0 ) time_index = 0
if (time_index > (num_time_stamps - 1)) time_index = num_time_stamps - 1

Pan_actual_time = line_time_stamp[time_index] - Pan_settle_time
                  - Pan_integration_time/2
                  + (pan_line - l0r_fill_pixels - 2*(time_index-1))*Pan_sample_time

Pan_nominal_time = Pan_actual_time
                   + (l0r_fill_pixels – nominal_fill_pixels) * Pan_sample_time
```

where:

- pan_line is the zero-referenced panchromatic line number (2N or 2N+1).
- nominal_fill_pixels is the amount of even/odd detector alignment fill to be inserted at the beginning of pixel columns that correspond to nominal detectors; that is, those detectors with a delay value of zero that are the basis for the Legendre polynomial LOS model. This value comes from the CPF.
- l0r_fill_pixels is the total amount of even/odd detector alignment fill to be inserted at the beginning of the pixel column associated with the current detector. It includes both the nominal_fill_pixels and the detector-specific delay fill required to align even/odd detectors. Note that these values will always be even for the panchromatic band.
- num_time_stamps is the total number of time codes (data frames) in the image. It is tested to ensure that time_index, the line_time_stamp index, does not go out of bounds.

- maximum_detector_delay is a constant offset that represents the largest amount of even/odd detector offset for any detector from the LOS model detector delay table. It is rounded to the nearest integer pixel because time offsets can only occur in whole line increments. The value of this parameter is not critical as the line-of-sight offsets computed for “maximum” detectors are divided by the maximum delay to compute offset-per-unit-delay coefficients. This parameter is set in a #define statement.
- Pan_settle_time is a small sample and hold time delay constant.

For the panchromatic band, the l0r_fill_pixels, nominal_fill_pixels, and maximum_detector_delay parameters are in units of panchromatic pixels.

Note that when fill is used to align even and odd detectors, the spatial difference between the nominal and actual look vectors is approximately compensated by the time difference between $t_{nominal}$ and t_{actual}.

4.2.1.6.2 Spacecraft Model

The spacecraft ephemeris and attitude models are constructed from the input preprocessed ancillary data by extracting the ancillary data that span the current image. Both ECI and ECEF versions of the ephemeris data are retained in the model structure to avoid the need to repeatedly invoke the ECI/ECEF coordinate system conversion. The ALIAS heritage roll-pitch-yaw representation of the attitude model is retained in the model structure, though a quaternion representation may be used in a future algorithm revision.

4.2.1.6.3 Prepare LOS Model Sub-Algorithm

This function gathers the information from the preprocessed ancillary data and the Level 0R/1R image data needed to process model data and run the LOS model.

The main steps are as follows:

1. Load the image time codes and convert to seconds since J2000 epoch (done by Ingest processing).
2. Determine the image time window.
3. Validate/smooth the image time codes.
4. Extract the multispectral and panchromatic integration times from the Level 0R/1R image frame header data.
5. Extract the associated ephemeris and attitude data from the preprocessed ancillary data stream.
6. Preprocess the input attitude data into a low-frequency stream, used for basic geometric modeling, and a high-frequency stream, used as a fine correction in the image resampler. This preprocessing was added to improve the ability of the geometric correction system to compensate for jitter disturbance frequencies above 10 Hz.

The input preprocessed ancillary data are stored in an HDF file. The attitude and ephemeris ancillary data streams each have an epoch time identifying the UTC date/time reference. Within these data streams, each attitude or ephemeris observation in the HDF file has a corresponding time offset relative to the epoch. This incoming ancillary data stream spans the entire imaging interval containing the image data represented in the Level 0R/1R input data. In creating the model, we identify and extract the ancillary data sequence required to process the current image data.

The input Level 0R/1R image data are also packaged in HDF files that include the image samples for each band and SCA, and the time codes assigned to each image line by the OLI instrument. These spacecraft time codes are provided by the OLI in CCSDS T-Field format, which includes days since epoch (16-bit integer), milliseconds of day (32-bit integer), and microseconds of millisecond (16-bit integer) fields:

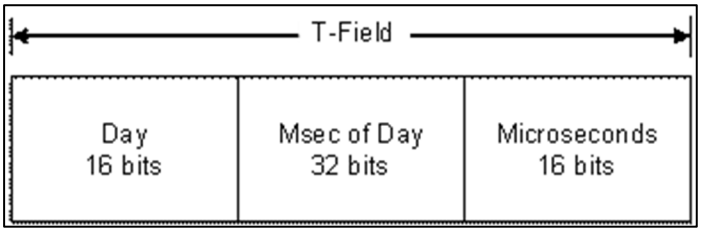

Figure 4-21. OLI Time Code Format

Level 0 (Ingest) processing will combine these raw time code fields, and adjust for the spacecraft epoch offset from J2000, to compute the time since the J2000 epoch in the form: days since J2000 epoch and seconds of day. Since they are derived from the spacecraft clock, the image time codes will be based on the same epoch used by the ancillary data (e.g., TAI seconds from J2000). Even though the initial time code conversion will occur in Level 0 processing, for completeness the processing described below begins with the raw time code fields shown in Figure 4-21.

4.2.1.6.3.1 Process Image Time Codes

The image time codes are loaded from the input HDF Level 0R/1R dataset. Even/odd detector alignment fill may be inserted into the Level 0R/1R imagery, as described above, so the image lines each contain samples collected at times that may be offset from the time specified by the corresponding time code. The relationship between these time codes, the OLI integration times, and the multispectral and panchromatic pixel center times has already been described above. The L0Rp data will contain one time code per multispectral image line, excluding fill, or a nominal 6701 time codes per scene. The image files themselves may be up to 10 lines longer to accommodate the even/odd detector alignment fill.

A defect in the OLI timing logic can lead to erroneous time codes being generated when the microseconds or milliseconds fields fail to roll over properly. In the first case, the microseconds field can reach 1000 and increment the millisecond field without rolling over to zero. In the second case, the milliseconds field can reach 86400000 and

increment the day field without rolling over to zero. Though these errors should be detected and corrected during Level 0 processing, the following time code validation logic will detect and correct this effect, as well as other suspicious time codes.

1. Convert the time codes to seconds from spacecraft epoch:
   Line_time = TC_Day*86400 + TC_MSec/1000 + TC_Micro/1e6
   Note that an IEEE 754 double precision (64-bit) number with a 52-bit fraction should provide sufficient precision to represent time differences from 01JAN2000 to 01JAN2050 with microsecond accuracy (1.6e15 microseconds < 2^51).
2. Validate the image time codes as follows:
   a. Loop through the time codes from 1 to N-1, where N is the number of image data frames/time codes, and test the difference between the current and previous time codes against the nominal frame time from the CPF using the #define tolerance DTIME_TOL. The first of two consecutive time codes that are within the tolerance is the first valid time code.
   b. Initialize the OLI clock model by setting the least squares variables to zero: $A_{00} = A_{01} = A_{11} = L_0 = L_1 = 0$
      i. Since the normal equation matrix, A, is symmetric, $A_{10} = A_{01}$ so it is not computed separately.
      ii. Add the first valid time code observation by adding 1 to A_{00}. This is all that is required since, by definition, the index difference and time difference (see below) are zero at the first valid point.
   c. For each subsequent time code:
      i. Compare the time difference from the previous time code to the nominal value using the DTIME_TOL threshold.
      ii. If a time code fails this check, see if the special conditions of the known OLI time code defect apply:
         1. If the time code difference deviates from the nominal value by more than 0.5 milliseconds:
            a. If the time code microseconds field = 1000, subtract 1000.
            b. If the time code milliseconds field = 86400000 and the microseconds field = 0, set the milliseconds field to zero.
            c. Apply the spacecraft clock epoch offset from J2000 to the updated time code.
            d. Recalculate the time code difference.
      iii. Compare the time code difference to a larger outlier tolerance (OUTLIER_TOL) chosen to bound the possible drift in the OLI clock relative to the spacecraft clock (currently set to 50 microsec).
      iv. If the time code difference is within the outlier range, add the current time to a least squares linear OLI clock model:
         1. Δnum = current index – first valid index
         2. Δtime = current time – first valid time
         3. Accumulate:
            a. Valid point count: A_{00} += 1

b. Index difference: A_{01} += Δnum
c. Squared index diff: A_{11} += Δnum*Δnum
d. Time difference: L_0 += Δtime
e. Time diff*index diff: L_1 += Δnum*Δtime

d. Once all time codes have been analyzed, solve for the linear OLI clock model parameters:
   i. determinant = $A_{00}*A_{11} - A_{01}*A_{01}$
   ii. If abs(determinant) <= 0.0, return an error
   iii. Offset = first valid time + ($A_{11}*L_0 - A_{01}*L_1$) / determinant
   iv. Rate = ($A_{00}*L_1 - A_{01}*L_0$) / determinant
e. Use the correction model to replace bad time codes:
   i. For each time code:
      1. Calculate the corresponding model time as:
         Mtime = Offset + (current index – first valid index) * Rate.
      2. Calculate the actual time – model time difference.
         Diff = abs(time code – Mtime)
      3. Test the difference against DTIME_TOL.
      4. If the difference exceeds DTIME_TOL, replace the current time code with the model value, Mtime.
f. If no valid time codes were found, return an error.
g. Calculate the average observed frame time, delta_time, by subtracting the first valid/corrected time code from the last valid/corrected time code and dividing by the number of time code minus one.
h. Store delta_time (MS frame time) and delta_time/2 (pan frame time) in the model.

3. Set the image start time: image_start = line_time[0].
4. Subtract the image start time from the line time codes so that the times are seconds from the image start.
5. Store the image start UTC epoch (image_year, image_day, image_seconds) and the image line offset times in the model structure.
6. Report/trend the results of the time code processing, including the following:
   a. WRS Path/Row (input parameters)
   b. Image UTC epoch (year, day, seconds of day)
   c. L0R ID (input parameter)
   d. Work order ID (input parameter)
   e. Computed frame time (delta_time)
   f. Number of replaced time codes (bad_image_time_count)
7. Check the pan and MS detector integration times in the L0Rp frame header and if they are valid (> 0), convert them to units of seconds and load them in the model. Otherwise, use the nominal values from the CPF converted to units of seconds.
8. Load the detector settling times from the CPF into the model after converting them to units of seconds.

4.2.1.6.3.2 Extract Ancillary Ephemeris and Attitude Data

The subset of ancillary ephemeris and attitude data needed to span the image data are extracted from the Level 0R data by the ancillary data preprocessing algorithm. The logic to do the required subsetting is reiterated below for reference. These data are read from the input preprocessed ancillary data stream and stored in the model structure during model creation.

The ephemeris data extraction/subsetting procedure is as follows:

1. Compute the time offset from the ephemeris epoch time to the desired ephemeris start time for this image.
   ephem_start = image_seconds – ancillary_overlap – ephem_seconds
   Noting that image_seconds and ephem_seconds are the seconds of day fields from the image and ephemeris epoch times, respectively, and ancillary_overlap is the desired extra ancillary data before and after the image window (set in a #define statement).
2. Loop through the ephemeris sample times to find the last entry that does not exceed ephem_start. This is the ephemeris start index (eph_start_index).
3. Compute the time offset from the ephemeris epoch time to the desired ephemeris stop time for this image.
   ephem_stop = image_seconds + line_time[N-1] + ancillary_overlap – ephem_seconds
4. Loop through the ephemeris sample times to find the first entry that exceeds ephem_stop. This is the ephemeris stop index (eph_stop_index).
5. Compute a new ephemeris UTC epoch for this image:
   imgeph_year = ephem_year
   imgeph_day = ephem_day
   imgeph_seconds = ephem_seconds + ephem_samp_time[eph_start_index]
6. Load the ECI and ECEF ephemeris samples from eph_start_index to eph_stop_index (inclusive) into the preprocessed ancillary data output, adjusting the sample times so that they are offset from the UTC epoch computed in step 5.

The attitude data extraction/subsetting procedure is as follows:

1. Compute the time offset from the attitude epoch time to the desired attitude start time for this image.
   att_start = image_seconds – ancillary_overlap – att_seconds
   Noting that image_seconds and att_seconds are the seconds of day fields from the image and attitude epoch times, respectively.
2. Loop through the attitude sample times to find the last entry that does not exceed att_start. This is the attitude start index (att_start_index).
3. Compute the time offset from the attitude epoch time to the desired attitude stop time for this image.
   att_stop = image_seconds + line_time[N-1] + ancillary_overlap – att_seconds

4. Loop through the attitude sample times to find the first entry that exceeds att_stop. This is the attitude stop index (att_stop_index).
5. Compute a new attitude UTC epoch for this image:
   imgatt_year = att_year
   imgatt_day = att_day
   imgatt_seconds = att_seconds + att_samp_time[att_start_index]
6. For Earth-view acquisitions, load the roll-pitch-yaw samples from att_start_index to att_stop_index (inclusive) into the preprocessed ancillary data output, adjusting the sample times so that they are offset from the UTC epoch computed in step 5.
7. For lunar/stellar acquisitions, convert the ECI quaternion samples from att_start_index to att_stop_index (inclusive) to ECI roll-pitch-yaw values, as described below, and store the computed roll-pitch-yaw values in the output, adjusting the sample times so that they are offset from the UTC epoch computed in step 5.

4.2.1.6.3.3 Converting ECI Quaternions to Roll-Pitch-Yaw

For lunar and stellar acquisitions, the ECI attitude representation is stored in the model structure. In the baseline model, this is done by converting the ECI quaternions to roll-pitch-yaw values relative to the ECI axes. This is one of the motivations for considering a transition to using a quaternion attitude representation in the model in the future.

The ECI quaternions are converted to roll-pitch-yaw values as follows:

1. Compute the rotation matrix corresponding to the ECI quaternion values:

$\mathbf{M}_{ACS2ECI} =$

$$\begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2 & 2(q_1q_2 - q_3q_4) & 2(q_1q_3 + q_2q_4) \\ 2(q_1q_2 + q_3q_4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 - q_1q_4) \\ 2(q_1q_3 - q_2q_4) & 2(q_2q_3 + q_1q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix}$$

2. Compute the corresponding ACS to ECI roll-pitch-yaw values:

$$\text{roll}' = -\tan^{-1}\left(\frac{M_{2,1}}{M_{2,2}}\right)$$

$$\text{pitch}' = \sin^{-1}\left(M_{2,0}\right)$$

$$\text{yaw}' = -\tan^{-1}\left(\frac{M_{1,0}}{M_{0,0}}\right)$$

Note that in implementing these calculations, it is important to use the ATAN2 rather than the ATAN arctangent implementation in order to retain the correct quadrants for the Euler angles. This is not a concern in Earth-view imagery where the angles are always small, but becomes an issue for these lunar/stellar ACS to ECI angles.

3. Store the ECI roll-pitch-yaw values in the model attitude data table.

4.2.1.6.3.4 Jitter Correction Data Preprocessing

Jitter correction preprocessing operates on the roll-pitch-yaw attitude data stream extracted from the spacecraft ancillary data to separate the low-frequency spacecraft pointing effects from the higher-frequency jitter disturbances. The low-frequency pointing model is used for line-of-sight projection and other geolocation processing, while the high-frequency jitter effects are applied as per-line corrections during image resampling. To implement this frequency separation in the line-of-sight model, the original attitude sequence is passed through a low-pass filter with a cutoff frequency defined as a parameter in the CPF. This cutoff frequency will nominally be in the 1 Hz to 10 Hz range. The value ultimately selected for this cutoff frequency will depend on the actual disturbance profile observed in the spacecraft attitude data. The high-frequency data stream should be limited in magnitude to subpixel (ideally sub-half-pixel) effects, but the lower the cutoff frequency can be, the sparser (and smaller) the OLI resampling grid can be made in the line (time) dimension.

The low-pass filtered version of the attitude sequence is differenced with the original data to construct the complementary high-pass data sequence. The high-pass sequence is then interpolated at the image line times for the OLI panchromatic band to provide a table containing high-frequency roll-pitch-yaw corrections for each image line. This jitter table is stored in the OLI line-of-sight model. The original attitude sequence in the line-of-sight model is replaced with the low-pass filtered sequence to avoid double counting the high-frequency effects. Figure 4-22 depicts this process.

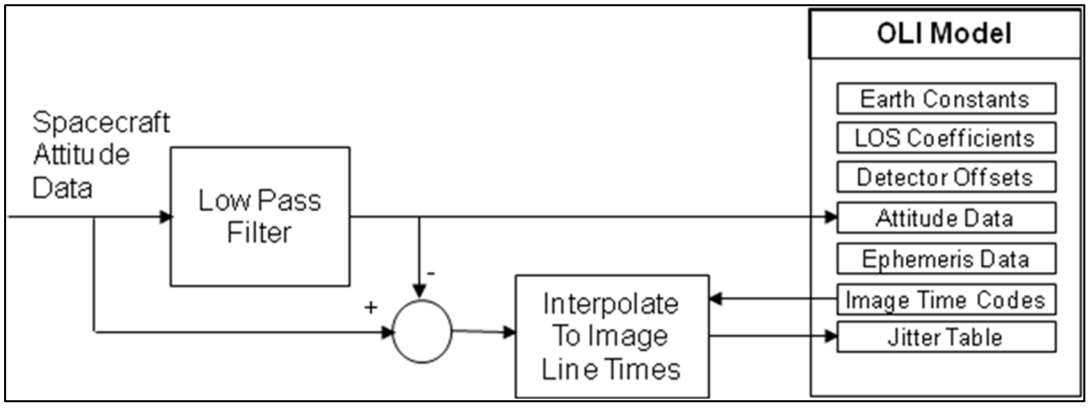

Figure 4-22. Jitter Correction Table Generation Data Flow

The jitter table construction processing sequence is as follows:

1. Extract a copy of the original attitude data sequence from the OLI line-of-sight model.
2. Retrieve the low-pass filter cutoff frequency from the CPF.
3. Design a low-pass filter with the desired cutoff frequency and apply it to the attitude data.

a. Use the cutoff frequency and attitude data sampling time to compute the size of the desired filter, as follows:
    i. Compute the normalized cutoff frequency (the ratio of the cutoff frequency to the attitude data sampling frequency):
        n_cutoff = cutoff_frequency / attitude_sample_frequency
        Note that this is the same as:
        n_cutoff = cutoff_frequency * attitude_sample_time
    ii. Compute the number of samples per cycle at the cutoff frequency:
        Nsamp = 1 / n_cutoff
    iii. Multiply the number of samples per cycle by 3 and add 1 to yield the desired filter size:
        FSize = 3*Nsamp + 1
    iv. If this results in an even filter size, add one:
        If (FSize modulo 2 == 0) FSize = FSize + 1
b. Use the Remez exchange algorithm to design the filter and generate the filter weights. The standard Parks-McClellan Finite Impulse Response (FIR) digital filter design method uses the Remez exchange algorithm (ref. Theory and Application of Digital Signal Processing, Rabiner and Gold, Prentice-Hall, 1975). A C implementation of this algorithm called remez.c, authored by Jake Janovetz at the University of Illinois, is available as shareware. This implementation specifies the desired (low pass, in this case) filter response using the following parameters:
    i. Filter size (number of taps) – FSize computed in item a. above.
    ii. Number of frequency bands to use – 2, one pass band (low frequency) and one stop band (high frequency).
    iii. Band frequency bounds – 0 to the normalized cutoff frequency (n_cutoff) for the pass band and 1.5*n_cutoff to 0.5 (normalized Nyquist frequency) for the stop band.
    iv. Desired band gains – 1 for pass band (low) and 0 for stop band (high).
    v. Band weights (how tightly to constrain the actual filter response to the design filter response in each band) – 1 for pass band and 10 for stop band.
    vi. Filter type – BANDPASS (the low-pass filter is a special case of the more general BANDPASS filter type supported by the remez algorithm.
c. Make sure the synthesized filter is normalized (weights sum to 1) by adding the filter tap values and dividing each tap by the total.
    sum = Σ h[i] where h[i] are the FSize filter taps.
    h'[i] = h[i] / sum for i = 1 to FSize.
d. Convolve the filter with the roll-pitch-yaw attitude data one axis at a time:
    half_size = FSize / 2
    for index = 0 to num_rpy – 1
        low_roll[index] = low_pitch[index] = low_yaw[index] = 0
        for ii = -half_size to half_size
            if (index + ii < 0) j = -index – ii
            else if (index + ii < num_rpy) j = index + ii
            else j = 2*num_rpy – index - ii – 1

```
                    low_roll[index] += roll[j]*h[ii+half_size]
                    low_pitch[index] += pitch[j]*h[ii+half_size]
                    low_yaw[index] += yaw[j]*h[ii+half_size]
```

4. Subtract the low-pass filtered sequences from the original sequences to extract the high-frequency portion of the data, and transfer any residual bias (non-zero mean value) from the imaging portion of the high-frequency sequence to the low-frequency sequence:

```
roll_bias = pitch_bias = yaw_bias = 0
att_pts = 0
for index = 0 to nrpy–1
    high_roll[index] = roll[index] – low_roll[index]
    high_pitch[index] = pitch[index] – low_pitch[index]
    high_yaw[index] = yaw[index] – low_yaw[index]
    if ( image_start_time < attitude_time[index] < image_stop_time )
        roll_bias += high_roll[index]
        pitch_bias += high_pitch[index]
        yaw_bias += high_yaw[index]
        att_pts += 1
roll_bias = roll_bias / att_pts
pitch_bias = pitch_bias / att_pts
yaw_bias = yaw_bias / att_pts
for index = 0 to nrpy-1
    high_roll[index] -= roll_bias
    low_roll[index] += roll_bias
    high_pitch[index] -= pitch_bias
    low_pitch[index] += pitch_bias
    high_yaw[index] -= yaw_bias
    low_yaw[index] += yaw_bias
```

5. Interpolate the high-frequency sequence values at the panchromatic band line sampling times to create the model jitter table:

```
For each panchromatic image line = 0 to number of pan lines:
    Compute the line sampling time as the following:
        index = line / 2
        pan_time = line_time_stamp[index] - pan_settle_time
                 - pan_integration_time/2
                  + (line - 2*time_index)*pan_sample_time
    Convert to time from attitude epoch:
        pan_time += image_epoch – attitude _epoch
    Interpolate high frequency roll-pitch-yaw values at this time using
    four-point Lagrange interpolation:
        Compute starting index for interpolation:
        index = floor(pan_time / attitude_sample_time) – 1
        Compute the fractional sample offset to the pan line time:
        w = pan_time / attitude_sample_time – index – 1
        Compute the Lagrange weights:
        w1 = -w * (w – 1) * (w – 2) / 6
```

w2 = (w + 1) * (w – 1) * (w – 2) / 2
w3 = -w * (w + 1) * (w – 2) / 2
w4 = (w + 1) * w * (w – 1) / 6
Interpolate:
roll = high_roll[index]*w1 + high_roll[index+1]*w2
+ high_roll[index+2]*w3 + high_roll[index+3]*w4
pitch = high_pitch[index]*w1 + high_pitch[index+1]*w2
+ high_pitch[index+2]*w3 + high_pitch[index+3]*w4
yaw = high_yaw[index]*w1 + high_yaw[index+1]*w2
+ high_yaw[index+2]*w3 + high_yaw[index+3]*w4

6. Replace the original model attitude data sequence with the low-pass filtered attitude data sequence.

At the completion of this sub-algorithm, the model structure contains the image frame time stamps, the multispectral and panchromatic sample, integration, and settling times, the ancillary ephemeris data, in both ECI and ECEF representations, covering the image, and the ancillary attitude data covering the image

4.2.1.6.4 Process LOS Model Sub-Algorithm

This function loads the LOS Legendre polynomial coefficients and other model components from the CPF, and performs additional processing on the attitude and ephemeris information in the LOS model structure. It invokes the following sub-algorithms.

4.2.1.6.4.1 Read CPF Model Parameters Sub-Algorithm

This function loads model components from the CPF. These CPF elements are included in the OLI model to make it self-contained for purposes of line-of-sight computations.

Key CPF parameters loaded into the geometric model include the following:

- Earth orientation parameters – the UT1UTC and pole wander (x,y) parameters for the current day are stored in the model to avoid the necessity of repeatedly looking them up in the CPF. WGS84 ellipsoid parameters (semi-major and semi-minor axes and eccentricity) are also extracted from the CPF, as are physical constants such as the Earth rotation velocity and the speed of light.
- OLI offset from spacecraft center of mass – a 3-vector that captures the small offset, in spacecraft body coordinates, between the OLI instrument, where images are captured, and the spacecraft center of mass, the position of which is reported in the ancillary ephemeris data, making it possible to translate the ephemeris data to the OLI. Technically, this would be the vector from the spacecraft center of mass to the center of the OLI entrance pupil. Note that this formulation assumes that the spacecraft on-board GPS data processing includes the GPS to spacecraft Center of Mass (CM) offset and that the spacecraft is, in fact, reporting CM positions, not GPS antenna positions. If the ephemeris represents the GPS antenna location, then we would need to know the spacecraft CM to GPS antenna offset as well.

- OLI to ACS alignment matrix – a 3-by-3 matrix that captures the relative orientation of the OLI coordinate system to the ACS coordinate system, making it possible to rotate the OLI instrument-space line-of-sight vectors into the ACS reference system. In the heritage ALIAS system, this was actually represented in the CPF by an ACS-to-instrument-rotation matrix, which was inverted for each LOS model invocation. Whichever convention is used in the CPF, the LOS model should store the OLI-to-ACS rotation matrix.
- OLI sensor parameters, including the nominal detector sampling rate, integration times (pan and MS), settling times (pan and MS), and Instantaneous Fields Of View (IFOVs), as well as the number of bands, SCAs per band, detectors per SCA, and nominal detector fill values.
- OLI line-of-sight Legendre polynomials – a set of 6 coefficients (3 along-track and 3 across-track) for each band on each SCA. Each set of 3 forms a 2nd order Legendre polynomial that is used to evaluate a nominal LOS angle (along- or across-track) for the detectors in that band on that SCA.
- OLI detector delay table – a table consisting of two values (along- and across-track) per detector reflecting the offset of each actual detector from its nominal location (as modeled by the 2nd order Legendre polynomials – see below). This table contains the combination of the even/odd detector offsets, any offsets due to detector deselect (i.e., the operational use of a detector from one of the redundant rows), and the small subpixel offsets due to non-ideal detector placement that are applied in the image resampling procedure. This table is therefore needed in those LOS projection algorithms that use either actual (whole pixel offsets) or exact (full subpixel offsets) detector locations as well as in the image resampling algorithm.

4.2.1.6.4.2 Read LOS Vectors Sub-Algorithm

This function retrieves the line-of-sight vectors from the CPF. The line-of-sight vectors are stored as sets of 2nd order Legendre polynomial coefficients. There is a unique set of 6 coefficients for each band of each SCA, 3 for the along-track polynomial and 3 for the across-track polynomial. These values are read from the CPF and stored in the LOS model. The polynomials are used to compute along- and across-track viewing angles for each nominal detector.

4.2.1.6.4.3 Initialize the Precision Model Sub-Algorithm

This function initializes the precision LOS correction model parameters. If the optional precision model input parameters are provided, those values are used. In the normal case, those parameters are absent and the correction model is initialized as follows:

Set the precision correction reference time to the beginning of the scene:
t_ref = 0.0
Set the ephemeris correction model order to zero: eph_order = 0
Set both ephemeris X correction parameters to zero:
x_corr[0] = 0.0, x_corr[1] = 0.0
Set both ephemeris Y correction parameters to zero:
y_corr[0] = 0.0, y_corr[1] = 0.0

Set both ephemeris Z correction parameters to zero:
z_corr[0] = 0.0, z_corr[1] = 0.0
Set the attitude correction model order to zero: att_order = 0
Set all three attitude roll correction parameters to zero:
roll_corr[0] = 0.0, roll_corr[1] = 0.0, roll_corr[2] = 0.0
Set all three attitude pitch correction parameters to zero:
pitch_corr[0] = 0.0, pitch_corr[1] = 0.0, pitch_corr[2] = 0.0
Set all three attitude yaw correction parameters to zero:
yaw_corr[0] = 0.0, yaw_corr[1] = 0.0, yaw_corr[2] = 0.0

Note that these parameters are used to compute the corrected ephemeris and attitude data sequences, which are also stored in the model. The parameters themselves are included in the model primarily to document the magnitude of the corrections applied and to facilitate more advanced uses of the model creation logic. For example, it is sometimes useful to be able to force a particular model bias (e.g., a roll angle) into a model that is to be used for data simulation. Therefore, though not strictly necessary for operational data processing, these parameters aid in anomaly resolution, data simulation, and algorithm development. In normal operations, these initial correction parameters are all zero and the "corrected" attitude and ephemeris data sequences are identical to the "original" attitude and ephemeris data prior to the execution of the LOS model correction algorithm. Subsequent algorithms (e.g., LOS projection) operate on the corrected data.

4.2.1.6.4.4 Correct Attitude Sub-Algorithm

This function applies the ACS/body space attitude corrections computed by the LOS/precision correction procedure to the attitude data sequence. It outputs a parallel table of roll-pitch-yaw values with the precision corrections applied. In the model creation context, the precision corrections are zero, so the two sets of attitude data are identical. Although applying the precision corrections to construct the corrected attitude sequence could be said to be overkill for model creation (since the corrections are nominally zero at this point), this capability is required for LOS model correction and is used here to support the use of the model creation algorithm for data simulation and anomaly resolution, as it makes it possible to force initial biases into the model. This sub-algorithm will also be used by the LOS/precision correction algorithm to create the precision model. Note that the formulation is somewhat different for Earth-view scenes (Acquisition Type = Earth) than it is for lunar and stellar observations.

4.2.1.6.4.4.1 Earth Scenes

For Earth-view scenes, the sequence of transformations required to convert a line-of-sight in the OLI instrument coordinate system, generated using the Legendre polynomials, is as follows:

$$\underline{\mathbf{x}}_{ECEF} = \mathbf{M}_{ORB2ECEF}\,\mathbf{M}_{ACS2ORB}\,\mathbf{M}_{Precision}\,\mathbf{M}_{OLI2ACS}\,\underline{\mathbf{x}}_{OLI}$$

where: $\underline{\mathbf{x}}_{OLI}$ is the Legendre-derived instrument LOS vector
$\mathbf{M}_{OLI2ACS}$ is the OLI-to-ACS alignment matrix from the CPF

$\mathbf{M}_{Precision}$ is the correction to the attitude data computed by the LOS/precision correction procedure
$\mathbf{M}_{ACS2ORB}$ is the spacecraft attitude (roll-pitch-yaw)
$\mathbf{M}_{ORB2ECEF}$ is the orbital to ECEF transformation computed using the ECEF ephemeris
$\underline{\mathbf{x}}_{ECEF}$ is the LOS vector in ECEF coordinates

Note that in the heritage ALIAS implementation the sequence was as follows:

$$\underline{\mathbf{x}}_{ECEF} = \mathbf{M}_{ORB2ECEF}\ \mathbf{M}_{Precision}\ \mathbf{M}_{ACS2ORB}\ \mathbf{M}_{OLI2ACS}\ \underline{\mathbf{x}}_{OLI}$$

For nadir-viewing imagery, the $\mathbf{M}_{ACS2ORB}$ matrix is close to the identity matrix, so there is little difference. Since OLI will occasionally be viewing off-nadir and it is more natural to model attitude errors in the ACS/body coordinate system, the order is reversed for Landsat 8/9. The impact is minimal in the model and LOS projection, but becomes more important for the LOS/precision correction algorithm.

This new sub-algorithm pre-computes the $\mathbf{M}_{ACS2ORB}\ \mathbf{M}_{Precision}$ combination and stores the corresponding corrected roll-pitch-yaw attitude sequence in the model structure. This approach has the following advantages:

- It streamlines the application of the model for LOS projection by removing the step of explicitly applying the precision correction.
- It allows for the use of a more complex correction model in the future since the application of the model is limited to this unit. Note that the Earth-view attitude correction model consists of the following model parameters:
  Precision reference time: t_ref in seconds from the image epoch (at the center of the image time window)
  Attitude model order: att_order = 2
  Roll bias and rate corrections: roll_corr[] = roll_bias, roll_rate
  Pitch bias and rate corrections: pitch_corr[] = pitch_bias, pitch_rate
  Yaw bias and rate corrections: yaw_corr[] = yaw_bias, yaw_rate
  The line-of-sight correction algorithm description describes this model in more detail.
- Retaining both the original and corrected attitude sequences in the model makes the model self-contained and will make it unnecessary for the LOS/precision correction algorithm to access the preprocessed ancillary data.

The disadvantage is that it doubles the size of the attitude data in the model structure.

The construction of the corrected attitude sequence proceeds as follows:

For each point in the attitude sequence j = 0 to K-1:
1. Compute the rotation matrix corresponding to the j^{th} roll-pitch-yaw values:

$\mathbf{M}_{ACS2ORB}$ =

$$\begin{bmatrix} \cos(p)\cos(y) & \sin(r)\sin(p)\cos(y)+\cos(r)\sin(y) & \sin(r)\sin(y)-\cos(r)\sin(p)\cos(y) \\ -\cos(p)\sin(y) & \cos(r)\cos(y)-\sin(r)\sin(p)\sin(y) & \cos(r)\sin(p)\sin(y)+\sin(r)\cos(y) \\ \sin(p) & -\sin(r)\cos(p) & \cos(r)\cos(p) \end{bmatrix}$$

2. Compute the precision correction at the time (t_att = att_seconds + att_time[j]) corresponding to the attitude sample:
   a. $roll\_correction = \sum_{i=0}^{att\_order-1} \mathbf{roll\_corr}[i] * (t\_att - t\_ref - image\_seconds)^i$
   b. $pitch\_correction = \sum_{i=0}^{att\_order-1} \mathbf{pitch\_corr}[i] * (t\_att - t\_ref - image\_seconds)^i$
   c. $yaw\_correction = \sum_{i=0}^{att\_order-1} \mathbf{yaw\_corr}[i] * (t\_att - t\_ref - image\_seconds)^i$

   Note that only the seconds of day fields are needed for the attitude and image epochs, as they are constrained to be based on the same year and day.
3. Compute the rotation matrix corresponding to roll_correction (r), pitch_correction (p), and yaw_correction (y) ($\mathbf{M}_{Precision}$), using the same equations presented in step 1 above.
4. Compute the composite rotation matrix: $\mathbf{M} = \mathbf{M}_{ACS2ORB}\ \mathbf{M}_{Precision}$.
5. Compute the composite roll-pitch-yaw values:

$$\text{roll}' = -\tan^{-1}\left(\frac{M_{2,1}}{M_{2,2}}\right)$$

$$\text{pitch}' = \sin^{-1}\left(M_{2,0}\right)$$

$$\text{yaw}' = -\tan^{-1}\left(\frac{M_{1,0}}{M_{0,0}}\right)$$

6. Store the composite roll'-pitch'-yaw' values in the j^{th} row of the corrected attitude data table.

4.2.1.6.4.4.2 Lunar and Stellar Scenes

For celestial (lunar or stellar) observations, the sequence of transformations required to convert a line-of-sight in the OLI instrument coordinate system, generated using the Legendre polynomials, is as follows:

$$\underline{\mathbf{x}}_{ECI} = \mathbf{M}_{ACS2ECI}\ \mathbf{M}_{Precision}\ \mathbf{M}_{OLI2ACS}\ \underline{\mathbf{x}}_{OLI}$$

where:
- $\underline{\mathbf{x}}_{OLI}$ is the Legendre-derived instrument LOS vector
- $\mathbf{M}_{OLI2ACS}$ is the OLI to ACS alignment matrix from the CPF
- $\mathbf{M}_{Precision}$ is the correction to the attitude data computed by the LOS/precision correction procedure
- $\mathbf{M}_{ACS2ECI}$ is the spacecraft attitude in the ECI frame derived from the ECI quaternions in the preprocessed ancillary data
- $\underline{\mathbf{x}}_{ECI}$ is the LOS vector in ECI coordinates

The advantage of modeling the precision attitude corrections in ACS rather than orbital coordinates becomes apparent here, since the orbital frame is not used in the lunar case.

This sub-algorithm pre-computes the $\mathbf{M}_{ACS2ECI}$ $\mathbf{M}_{Precision}$ combination and stores the corresponding corrected attitude sequence (as roll-pitch-yaw values relative to ECI) in the model structure. Another difference between the Earth-view and lunar/stellar models is in the formulation of the precision model. The lunar attitude correction model adds an acceleration term to the Earth-view correction model parameters:

> Precision reference time: t_ref in seconds from the image epoch (nominally near the center of the image time window)
> Attitude correction model order: att_order = 3
> Roll bias, rate, and acceleration corrections: roll_corr[] = roll_bias, roll_rate, roll_acceleration
> Pitch bias, rate, and acceleration corrections: pitch_corr[] = pitch_bias, pitch_rate, pitch_acceleration
> Yaw bias, rate, and acceleration corrections: yaw_corr[] = yaw_bias, yaw_rate, yaw_acceleration

Due to the different orders of the Earth-view and lunar correction models, this model is stored as an array in the model structure, along with a field defining the model order. The line-of-sight correction algorithm description describes the precision model in more detail.

The processing steps to construct the corrected attitude sequence are the same for lunar/stellar acquisitions, although the interpretation of the roll-pitch-yaw values is slightly different, and proceeds as follows:

> For each point in the attitude sequence j = 0 to K-1:
> 1. Compute the rotation matrix corresponding to the j^{th} ECI roll-pitch-yaw values:

$\mathbf{M}_{ACS2ECI}$ =

$$\begin{bmatrix} \cos(p)\cos(y) & \sin(r)\sin(p)\cos(y)+\cos(r)\sin(y) & \sin(r)\sin(y)-\cos(r)\sin(p)\cos(y) \\ -\cos(p)\sin(y) & \cos(r)\cos(y)-\sin(r)\sin(p)\sin(y) & \cos(r)\sin(p)\sin(y)+\sin(r)\cos(y) \\ \sin(p) & -\sin(r)\cos(p) & \cos(r)\cos(p) \end{bmatrix}$$

> 2. Compute the precision correction at the time (t_att = att_seconds + att_time[j]) corresponding to the attitude sample:
>   a. $roll\_correction = \sum_{i=0}^{att\_order-1} \mathbf{roll\_corr}[i] * (t\_att - t\_ref - image\_seconds)^i$
>   b. $pitch\_correction = \sum_{i=0}^{att\_order-1} \mathbf{pitch\_corr}[i] * (t\_att - t\_ref - image\_seconds)^i$

c. $$yaw\_correction = \sum_{i=0}^{att\_order-1} \mathbf{yaw\_corr}[i] * (t\_att - t\_ref - image\_seconds)^i$$

Note that only the seconds of day fields are needed for the attitude and image epochs as they are constrained to be based on the same year and day.

3. Compute the rotation matrix corresponding to roll_correction (r), pitch_correction (p), and yaw_correction (y):

$\mathbf{M}_{Precision}$ =

$$\begin{bmatrix} \cos(p)\cos(y) & \sin(r)\sin(p)\cos(y)+\cos(r)\sin(y) & \sin(r)\sin(y)-\cos(r)\sin(p)\cos(y) \\ -\cos(p)\sin(y) & \cos(r)\cos(y)-\sin(r)\sin(p)\sin(y) & \cos(r)\sin(p)\sin(y)+\sin(r)\cos(y) \\ \sin(p) & -\sin(r)\cos(p) & \cos(r)\cos(p) \end{bmatrix}$$

4. Compute the composite rotation matrix: $\mathbf{M} = \mathbf{M}_{ACS2ECI}\ \mathbf{M}_{Precision}$.
5. Compute the composite ACS to ECI roll-pitch-yaw values:

$$roll' = -\tan^{-1}\left(\frac{M_{2,1}}{M_{2,2}}\right)$$

$$pitch' = \sin^{-1}\left(M_{2,0}\right)$$

$$yaw' = -\tan^{-1}\left(\frac{M_{1,0}}{M_{0,0}}\right)$$

Note that in implementing these calculations, it is important to use the ATAN2 rather than the ATAN arctangent implementation in order to retain the correct quadrants for the Euler angles. This is not a concern in Earth-view imagery where the angles are always small, but becomes an issue for these lunar/stellar ACS to ECI angles.

6. Store the composite roll'-pitch'-yaw' values in the j^{th} row of the corrected attitude data table.

4.2.1.6.4.5 Correct Ephemeris Sub-Algorithm

This function applies the ephemeris corrections computed in the LOS/precision correction procedure to both the ECI and ECEF representations of the ephemeris stored in the model. Though applying the precision corrections to construct the corrected ephemeris sequence could be said to be overkill for model creation (since the corrections are nominally zero at this point), this capability is required for LOS model correction and is used here to support the use of the model creation algorithm for data simulation and anomaly resolution, as it makes it possible to force initial biases into the model. This sub-algorithm will also be used by the LOS/precision correction algorithm to create the precision model.

The precision correction parameters are stored in the LOS model in the spacecraft orbital coordinate system as three position (x_bias, y_bias, z_bias) corrections and three velocity (x_rate, y_rate, z_rate) corrections that, like the attitude corrections, are relative to t_ref. These values must be converted to the ECEF and ECI coordinate systems. Once the precision correction is determined in the ECEF/ECI coordinate system, the ECEF/ECI ephemeris values can be updated with the precision parameters.

Loop on LOS model ephemeris points j = 0 to N-1

Compute the precision correction:

Calculate delta time for precision correction:

dtime = ephem_seconds + ephem_time[j] – t_ref – image_seconds

Calculate the change in X, Y, Z due to precision correction. Corrections are in terms of spacecraft orbital coordinates.

dx orb = model precision x_corr[0] + model precision x_corr[1] * dtime
dy orb = model precision y_corr[0] + model precision y_corr[1] * dtime
dz orb = model precision z_corr[0] + model precision z_corr[1] * dtime

where:
model precision x_corr[0] = precision (orbital) update to X position
model precision y_corr[0] = precision (orbital) update to Y position
model precision z_corr[0] = precision (orbital) update to Z position
model precision x_corr[1] = precision (orbital) update to X velocity
model precision y_corr[1] = precision (orbital) update to Y velocity
model precision z_corr[1] = precision (orbital) update to Z velocity

Construct precision position and velocity “delta” vectors.

$$[\text{dorb}] = \begin{bmatrix} \text{dx orb} \\ \text{dy orb} \\ \text{dz orb} \end{bmatrix}$$

$$[\text{dvorb}] = \begin{bmatrix} \text{model precision x\_corr}[1] \\ \text{model precision y\_corr}[1] \\ \text{model precision z\_corr}[1] \end{bmatrix}$$

Calculate the orbit to ECF transformation [ORB2ECEF] using ECEF ephemeris (See the Ancillary Data Preprocessing Algorithm (Section 4.1.4) for this procedure).

Transform precision “delta” vectors to ECEF.

$$[\text{def}] = [\text{ORB2ECEF}][\text{dorb}]$$

$$[\text{dvef}] = [\text{ORB2ECEF}][\text{dvorb}]$$

Adjust ECEF ephemeris by the appropriate “delta” precision vector and store the new ephemeris in the model. These ephemeris points will be used when transforming an input line/sample to an output projection line/sample.

$$\text{modelef postion} = \text{ephemerisecef postion} + \text{decf}$$

$$\text{modelef velocity} = \text{ephemerisecef velocity} + \text{dvecf}$$

where:

All parameters are 3x1 vectors
ephemeris ecef values are the interpolated one-second ephemeris values in ECEF coordinates

Calculate the orbit to ECI transformation [ORB2ECI] using ECI ephemeris.

Transform precision “delta” vectors to ECI.

$$[\text{deci}] = [\text{ORB2ECI}][\text{dorb}]$$

$$[\text{dveci}] = [\text{ORB2ECI}][\text{dvorb}]$$

Adjust ECI ephemeris by the appropriate “delta” precision vector and store the new ephemeris in the model. These ephemeris points will be used with lunar/stellar observations.

$$\text{modeleci postion} = \text{ephemeriseci postion} + \text{deci}$$

$$\text{modeleci velocity} = \text{ephemeriseci velocity} + \text{dveci}$$

where:

All parameters are 3x1 vectors
ephemeris eci values are the interpolated one-second ECI ephemeris

Table 4-7 summarizes the contents of the LOS model structure. The estimated size of this structure is approximately 1.5 megabytes.

LOS Model Structure Contents
Satellite Number (8)
Format Version Number (for documentation and backward compatibility)
WRS Path
WRS Row (may be fractional)
Acquisition Type (Earth, Lunar, Stellar)
Earth Parameters
UT1UTC Correction (in seconds)
Pole Wander X Correction (in arc seconds)
Pole Wander Y Correction (in arc seconds)
WGS84 Ellipsoid Semi-Major Axis (in meters)
WGS84 Ellipsoid Semi-Minor Axis (in meters)
WGS84 Ellipsoid Eccentricity (dimensionless)
Earth Angular Velocity (radians/second)

LOS Model Structure Contents
Speed of Light (meters/second)
Image Model
Number of image lines
Image UTC epoch: image_year, image_day, image_seconds
For each line: frame time offset (in seconds) from image epoch
For each line: roll, pitch, yaw high frequency jitter correction (in radians)
Nominal alignment fill table (from CPF) one value per band per SCA (in pixels)
Detector alignment fill table (from L0R/L1R) one value per detector (in pixels)
Sensor Model
OLI to ACS reference alignment matrix [3x3]
Spacecraft center of mass to OLI offset in ACS reference frame [3x1] in meters
Integration Times (MS and pan) in seconds
Computed Sample Times (MS and pan) in seconds
Detector Settling Times (MS and pan) in seconds
Number of SCAs (14)
Number of Bands (9)
Along-Track IFOVs (MS and pan) in radians
Across-Track IFOVs (MS and pan) in radians
Number of Detectors per SCA Per Band (9x1 array)
Focal plane model parameters (Legendre coefs) [NSCAxNBANDx2x3] (in radians)
Detector delay table [NSCAxNBANDx2xNDET] (in pixels)
Ephemeris Model
Scene ephemeris data UTC epoch: imgeph_year, imgeph_day, imgeph_seconds
Number of ephemeris samples
Time from epoch (one per sample, nominally 1 Hz) (in seconds)
Original ECI position estimate (X, Y, Z) (one set per sample) (in meters)
Original ECI velocity estimate (Vx, Vy, Vz) (one set per sample) (in meters/sec)
Original ECEF position estimate (X, Y, Z) (one set per sample) (in meters)
Original ECEF velocity estimate (Vx, Vy, Vz) (one set per sample) (in meters/sec)
Corrected ECI position estimate (X, Y, Z) (one set per sample) (in meters)
Corrected ECI velocity estimate (Vx, Vy, Vz) (one set per sample) (in meters/sec)
Corrected ECEF position estimate (X, Y, Z) (one set per sample) (in meters)
Corrected ECEF velocity estimate (Vx, Vy, Vz) (one set per sample) (in meters/sec)
Attitude Model
Scene attitude data UTC epoch: imgatt_year, imgatt_day, imgatt_seconds
Number of attitude samples
Time from epoch (one per sample, nominally 50 Hz) (in seconds)
Original Roll, pitch, yaw estimate (one per sample) (in radians)
Corrected Roll, pitch, yaw estimate (one per sample) (in radians)
Precision Correction Model
Precision reference time (t_ref) seconds from image epoch
Ephemeris correction order: eph_order (0 none, 2 for Earth-view and lunar/stellar)
X correction model: x_bias, x_rate (meters, meters/sec)
Y correction model: y_bias, y_rate (meters, meters/sec)

LOS Model Structure Contents
Z correction model: z_bias, z_rate (meters, meters/sec)
Attitude correction order: att_order (0 none, 2 for Earth, 3 for lunar/stellar)
Roll correction model: roll_bias, roll_rate, roll_acc (rad, rad/sec, rad/sec^2)
Pitch correction model: pitch_bias, pitch_rate, pitch_acc (rad, rad/sec, rad/sec^2)
Yaw correction model: yaw_bias, yaw_rate, yaw_acc (rad, rad/sec, rad/sec^2)

Table 4-7. LOS Model Structure Contents

Note that in the precision correction model, only the first att_order correction model array elements are valid. For example, for Earth-view scenes att_order = 2 and roll_corr[0] = roll_bias, roll_corr[1] = roll_rate and roll_corr[2] is not used.

4.2.2 OLI Line-of-Sight Projection/Grid Generation Algorithm

4.2.2.1 Background/Introduction

The LOS projection and grid generation algorithm uses the OLI LOS model, created by the LOS model creation algorithm, to calculate the intersection of the projected lines-of-sight from selected OLI detector samples (pixels) with an Earth model (WGS84). The spacecraft position and pointing, OLI instrument alignment and offset information, and image timing data contained in the LOS model are used to construct the LOS for an individual OLI detector at a particular sample time. We then calculate the location where that line-of-sight intersects the Earth's surface, as defined by the WGS84 Earth ellipsoid or a specified elevation above or below that ellipsoid. LOS intersections for an array of detector samples that span each OLI SCA/FPM and spectral band are computed at the WGS84 ellipsoid surface, as well as at a range of elevation levels selected to span the actual terrain elevations found in the image area. The resulting array of projected lines-of-sight forms a three-dimensional grid of input (Level 1R) image pixel line/sample to output space (Level 1G) mappings that can be used to interpolate input/output pixel mappings for intermediate points. The resulting ability to rapidly compute input/output mappings greatly facilitates image resampling.

The LOS projection and grid generation algorithm can also work in an "inertial direction" mode in which the output space is in angular units, with respect to a set of reference inertial directions. This mode is used to process lunar data wherein the inertial coordinates (declination and right ascension) of the moon, computed from a planetary ephemeris, are used as the reference to define the output image frame. In this case, the lines-of-sight are computed in inertial coordinates, but are not projected to the Earth's surface.

Concerns about the temporal (line direction) grid density that would be required to adequately capture attitude deviations (jitter) at frequencies above 10 Hz motivated the addition of new grid functionality to support high-frequency image correction at image resampling time. Specifically, jitter sensitivity coefficients were added to each grid cell to allow the high-frequency attitude data in the OLI line-of-sight model jitter table to be converted to corresponding input image space line/sample offsets. These coefficients are used by the resampler to compute high-frequency line/sample corrections that refine

the output-to-input space image coordinate mappings provided by the grid. This allows the LOS grid to model only lower-frequency effects, making a sparser grid sampling in the time (line) direction possible.

Due to layout of the OLI focal plane, there are along-track offsets between spectral bands within each SCA, along-track offsets between even and odd SCAs, and a reversal of the band ordering in adjacent SCAs. This leads to an along-track offset in the imagery coverage area for a given band between odd and even SCAs, as well as an offset between bands within each SCA. To create more uniform image coverage within a geometrically corrected output product, the leading and trailing imagery associated with these offsets is trimmed (at image resampling time) based on image active area bounds stored in the grid.

4.2.2.2 Dependencies

The OLI LOS projection and grid generation algorithm assumes that the OLI LOS model creation algorithm has been executed to construct and store the OLI LOS model.

4.2.2.3 Inputs

The LOS projection and grid generation algorithm and its component sub-algorithms use the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs
ODL File (implementation)
CPF File Name
LOS Model File Name
DEM File Name
NOVAS Planetary Ephemeris File Name (for lunar processing)
Output Image Framing Parameters:
WRS Path for path-oriented scene framing (not necessarily the LOS model path)
WRS Row for path-oriented scene framing (not necessarily the LOS model row)
Map Projection (UTM, SOM, PS)
UTM Zone (use 0 to have code compute the zone)
Map Projection Parameters
Output Pixel Size(s)
Output Image Orientation
Frame Type (e.g., MINBOX)
Frame Bounds (e.g., corner coordinates, image size)
Grid Options:
Bands to Grid
CPF file contents
Thresholds and Limits (replaces System Table)
Grid Density (line/sample/height)
Default (WGS84) Spheroid parameter and Datum Codes
Scene framing band priority list
OLI LOS Model file contents (see LOS Model Creation Algorithm (4.2.1) for details)
WGS84 Earth Ellipsoid parameters
Earth Angular Velocity (rotation rate) in radians/second
PAN and MS settling times

Algorithm Inputs
Speed of light (in meters/second)
Acquisition Type (Earth, Lunar, Stellar)
OLI to ACS reference alignment matrix
Spacecraft CM to OLI offset in ACS reference frame (new)
Focal plane model parameters (Legendre coefs)
Detector delay table
Smoothed ephemeris at 1 second intervals (original and corrected)
Low pass filtered attitude history (original and corrected)
High frequency attitude perturbations (roll, pitch, yaw) per image line (jitter table)
Image time codes
Integration Time (MS and Pan)
OLI MS and pan detector settling times (msec)
Nominal detector alignment fill table
L0R detector alignment Fill Table
DEM file contents
Min and Max Elevation
NOVAS Planetary Ephemeris file contents
JPL Ephemeris Table (DE405) for celestial bodies (i.e., the moon) (see note 1)

4.2.2.4 Outputs

OLI Grid (see Table 4-8 and Table 4-9 below for detailed grid structure contents)
Grid Header (WRS path/row, acquisition type)
Output Image Framing Information (corner coordinates, map projection)
Image active area latitude/longitude bounds (for each band)
Grid Structure Information (number of bands/SCAs)
Grid Structures (one per SCA, per band)
Band number
Image dimensions (line/sample)
Pixel size
Grid cell size (image lines/samples per cell)
Grid dimensions (# rows/# columns/# Z-planes)
Z-plane zero reference and height increment
Arrays of input line/sample grid point coordinates
Arrays of output line and sample grid point mappings
Arrays of even/odd offset coefficients (2 per grid cell)
Arrays of forward (input/output) mapping polynomials (8 per grid cell per Z-plane)
Arrays of inverse (output/input) mapping polynomials (8 per grid cell per Z-plane)
Arrays of roll-pitch-yaw jitter line sensitivity coefficients (3 per grid cell per Z-plane)
Arrays of roll-pitch-yaw jitter sample sensitivity coefficients (3 per grid cell per Z-plane)
Rough mapping polynomials (one set per Z-plane)

4.2.2.5 Options

A NOVAS planetary ephemeris file (JPL DE405) must be provided when the Acquisition Type (in the LOS model) is Lunar.

4.2.2.6 Procedure

The LOS Projection algorithm uses the geometric LOS model created by the LOS Model Creation algorithm to relate OLI image pixels to ground locations, or in the case of lunar/stellar images, to ECI directions. The LOS model contains several components, including Earth orientation parameters, an image model (validated image time codes), a

sensor model, an ephemeris model, and an attitude model. The Level 1R image line/sample location is used to compute a time of observation (from the image model), a LOS vector (from the sensor model), the spacecraft position (from the ephemeris model) at the time of observation, and the spacecraft attitude (from the attitude model) at the time of observation. The LOS vector is projected to the Earth's surface, either the topographic surface at a specified elevation (e.g., derived from an input Digital Elevation Model), or the WGS84 ellipsoid surface, to compute the ground position associated with that Level 1R image location. This LOS projection procedure relating an input image location to an output ground location is referred to as the forward model. In image resampling, we typically need to find the Level 1R input space line/sample location corresponding to a particular Level 1G output space location so that the corresponding image intensity can be interpolated from the Level 1R data. This "inverse model" computation must be performed for every pixel in the output Level 1G product. To make this computation efficient, we create a table, or grid, of input/output mappings, parameterized by height, for use by the image-resampling algorithm. This algorithm description document describes both the forward model and grid generation procedures.

4.2.2.6.1 The Geometric Grid

The geometric grid provides a mapping from input Level 1R line/sample space to output Level 1G line/sample space. As such, it incorporates not only the sensor LOS to Earth intersection geometry captured by the forward model, but also the output image framing information, such as scene corners, map projection, pixel size, image orientation, and the bounds of the active image area for each band. The gridding procedure generates a mapping grid that defines a transformation from the instrument perspective (input space) to a user-specified output projection on the ground (output space). This output frame may be map-oriented (north-up) or path-oriented for Earth-view acquisitions. Celestial (lunar/stellar) acquisitions use an output frame based on inertial right ascension and declination coordinates. Once the frame is determined in output space, the input space is gridded. Then, the grid in input space is mapped to the output space, using the forward model. Transformation coefficients to transform a grid cell from input to output space are determined, as well as coefficients to transform a grid cell from output to input space.

The concept behind creating this resampling grid is to define only a sparse set of points for the relationship between an input line and sample location to output line and sample location (see Figure 4-23). Four grid points define a grid cell. A grid cell is defined as a rectangle in input space, but will be distorted when mapped to the output space. The sampling of points between grid cell points is chosen such that any two points defining a grid cell and a line in input space will map to a line in output space. Therefore, every grid cell defines a bilinear mapping between the input and output space, and vice versa. The method of only mapping and storing a small set of input points is much more efficient than trying to map points individually by invoking the LOS model for each point. This is especially the case since a rigorous implementation of the inverse model would have to be iterative.

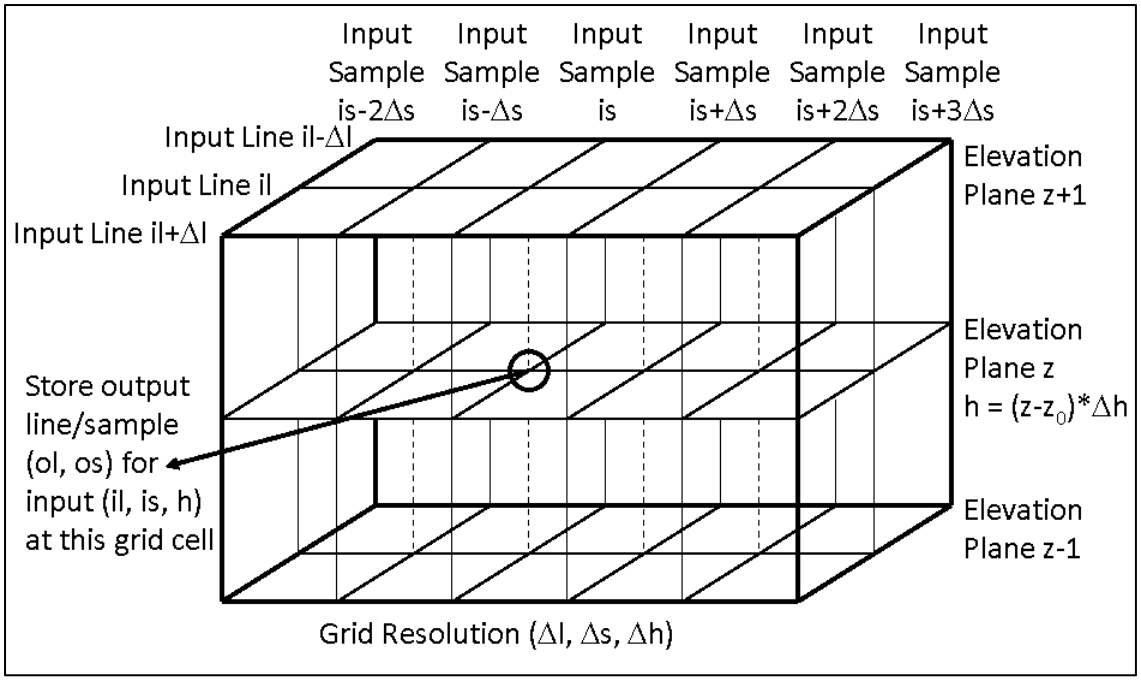

Figure 4-23. 3D Grid Structure

The 3D grid structure stores the output space line/sample coordinates corresponding to an array of input space line/sample/height coordinates.

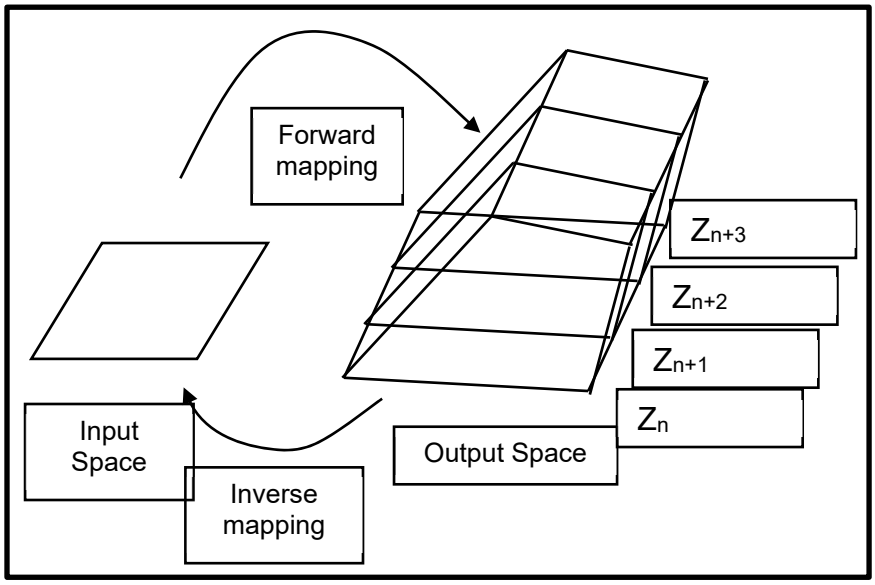

Figure 4-24. Forward and Inverse Mapping Using the Grid

The LOS projection grid contains projection information and three groups of mapping coefficients—one for mapping each grid cell from output space to input space (inverse), a second for mapping each grid cell from input space to output space (forward), and a third that gives an approximation or “rough” mapping of the entire output space to input

space. The first two mappings are described by a set of bilinear polynomials. The input space is represented by a line and sample location, while the output space is represented by a line and sample location, along with a Z component, where Z represents elevation. The output lines and samples can, in turn, be converted to X, Y projection space location by using the output image's upper-left projection coordinate and pixel size information in the grid header. Figure 4-24 shows how one input grid cell is mapped to a number of output grid cells, each grid cell representing a different elevation.

The number of grid cells is dependent on the line and sample size of each grid cell in the input image, elevation maximum, elevation minimum, and elevation increment. The input space is made up of evenly spaced samples and lines; values are associated with integer locations and can be indexed by an array of values: input_line[row] and input_sample[column]. Row refers to the index number, or row number, associated with the line spacing, while column refers to the index number, or column number, associated with the sample spacing. The output lines and samples typically do not fall on integer values (see Figure 4-25). This creates a two-dimensional array of indices for output line and sample locations. Adding elevation indices produces a three-dimensional array for output line and sample locations. The output lines and samples are then indexed by output_line[z][row][column] and output_sample[z][row][column], where Z refers to an elevation value. The row and column are the indices associated with the gridding of the raw input space. Since there is a mapping polynomial for each grid cell, the mapping polynomial coefficients are indexed by the same method as that used for output lines and samples; i.e., there are z*row*column sets of mapping coefficients.

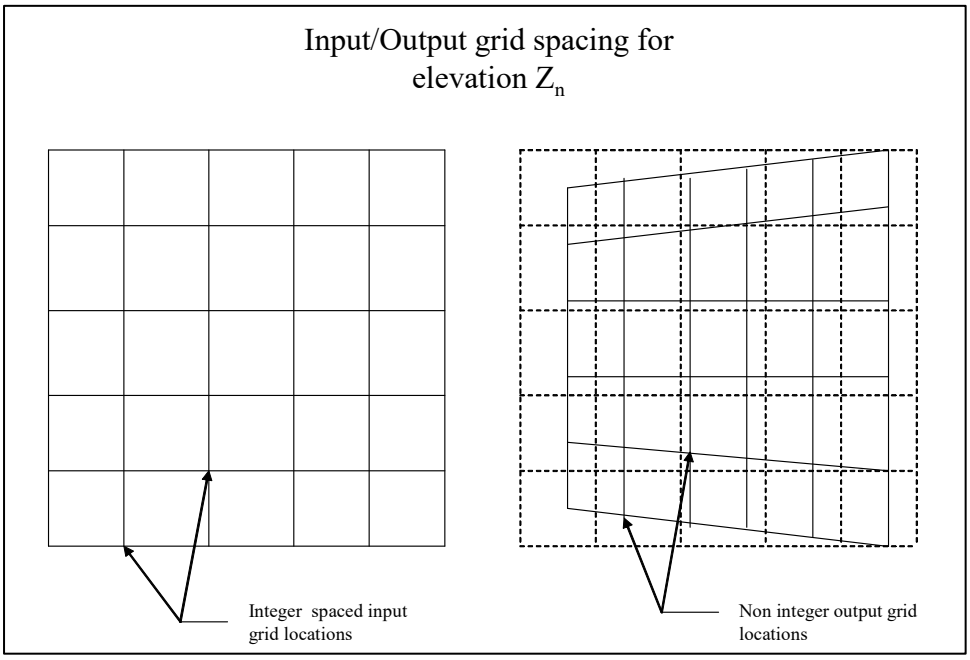

Figure 4-25. Mapping Integer Locations to "Non-integer" Locations

If a grid is being generated for a non-terrain-corrected image (i.e., no correction for relief is being applied), then the index for z is set such that $z_{elev=0}$ = zero elevation. Note that $z_{elev=0}$ does not necessarily have to be the first index in the array since there could be values for negative elevations. If the grid is being generated for a terrain-corrected image, then the indexes z_n and z_{n+1} are used such that the elevation belonging to the output location falls between the elevations associated with the indexes n and n+1. When performing an inverse mapping for a terrain-corrected image, two sets of input lines and samples are calculated from the polynomials for n and n+1. The actual input line and sample is interpolated between these lines and samples.

Example:

Output line/sample has r = row, c = col and z=n, n+1. If the inverse mapping coefficients are *a* and *b* for line and sample respectively, then:

$input\_line_n = bilinear(a_n, output\_line, output\_sample)$
$input\_sample_n = bilinear(b_n, output\_line, output\_sample)$
$input\_line_{n+1} = bilinear(a_{n+1}, output\_line, output\_sample)$
$input\_sample_{n+1} = bilinear(b_{n+1}, output\_line, output\_sample)$

bilinear is the bilinear mapping function (described below) for each grid cell.

If *e* is the elevation for the output line and sample location, then the weights used to interpolate between the two input line/sample locations are as follows:

$$w_n = \frac{e_{n+1} - e}{e_{n+1} - e_n} \qquad w_{n+1} = \frac{e - e_n}{e_{n+1} - e_n}$$

e_n, e_{n+1} and *e* are the elevations associated with z_n , z_{n+1} , and the output line and sample, respectively.

The final line/sample location is found from the following:

$input\_line = w_n * input\_line_n + w_{n+1} * input\_line_{n+1}$
$input\_sample = w_n * input\_sample_n + w_{n+1} * input\_sample_{n+1}$

The grid must contain a zero elevation plane. If the input minimum elevation is greater than zero, it is set to zero. If the input maximum elevation is less than zero, it is set to zero.

Given the elevation maximum, minimum, and increment, determine the number of z planes and the index of the zero elevation plane. Adjust the minimum and maximum elevations to be consistent with the elevation increment.

The number of z planes is determined from the following:

$$\text{number of z planes} = (\text{int})\left(\text{ceil}\left(\frac{\text{elevation maximum}}{\text{elevation increment}} \right) - \text{floor}\left(\frac{\text{elevation minimum}}{\text{elevation increment}} \right) \right) + 1$$

The plane for an elevation of zero is then found at:

$$z_{elev=0} = -\text{floor}\left(\frac{\text{elevation minimum}}{\text{elevation increment}} \right)$$

The new minimum and maximum elevation due to the values calculated above are as follows:

elevation minimum = $-z_{elev=0}$ * (elevation increment)

elevation maximum = (number of z planes – 1 – $z_{elev=0}$) * (elevation increment)

4.2.2.6.2 LOS Projection/Grid Generation Procedure Overview

The LOS Projection/Grid Generation procedure is executed in the following five stages:

1. Data Input - First, the required inputs are loaded. This includes reading the processing parameters from the input ODL parameter file, loading the LOS model from its HDF file, reading static gridding parameters from the CPF, and loading the elevation data from the DEM.
2. Scene Framing - The parameters of the output image space are computed based on the scene-framing scheme specified in the input ODL file. This includes calculating bounds for the active image area that excludes the leading and trailing SCA imagery, and using one of several available methods for determining the Level 1G scene corners. The scene-framing parameters are stored in the grid structure for eventual inclusion in the geometric metadata for the Level 1G product.
3. Grid Definition - The grid parameters are established to ensure adequate density in the space (sample), time (line), and elevation (z-plane) dimensions. The required data structures are allocated and initialized.
4. Grid Construction - The forward model is invoked for each grid intersection to construct the array of input space to output space mappings. A separate grid structure is created for each SCA and each band. The grid mapping polynomial coefficients are computed from the input space to output space mapping results for each grid cell. Once the basic grid mappings are defined, the forward model is invoked with small attitude perturbations about each axis in order to evaluate the sensitivity of the input space to output space mapping to small attitude deviations. The resulting sensitivity coefficients are stored with each grid cell for subsequent use in computing high-frequency jitter corrections during image resampling. Figure 4-26 shows a high-level data flow for the creation and use of these sensitivity coefficients.
5. Finalize and Output Grid - Derived grid parameters, such as the global rough mapping coefficients, are added to the grid structure, and the entire structure is written to a disk file. This also includes evaluating the small, but significant,

parallax effects caused by the time delay between when adjacent even and odd detectors sample the same along-track location. These effects are modeled in the grid as along- and across-track sensitivity coefficients that are scaled by the output point elevation and the even/odd detector offset, which can vary by pixel for OLI (due to detector deselect), rather than by band.

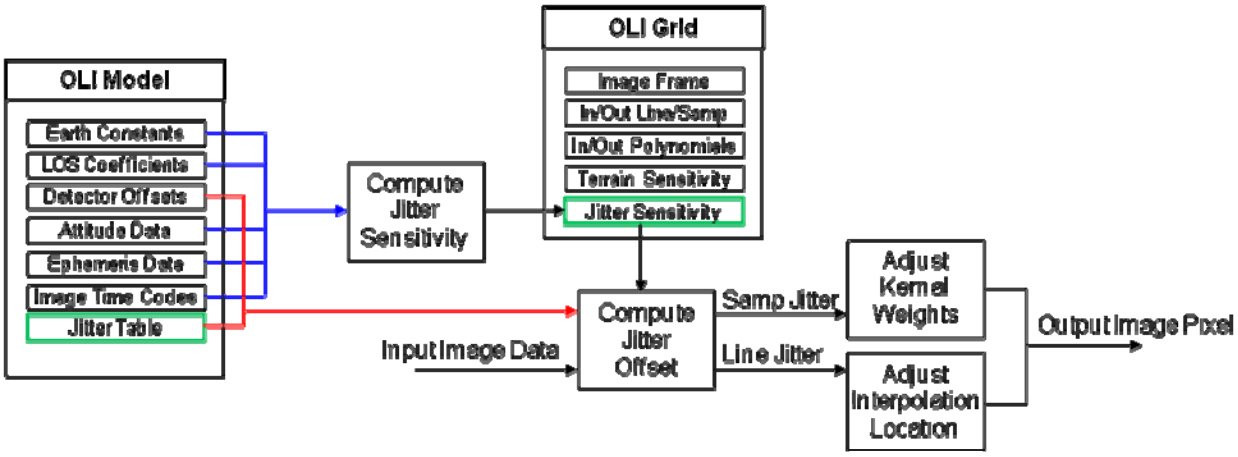

Figure 4-26. Jitter Correction Data Flow

Figure 4-27 shows a block diagram for the LOS Projection algorithm.

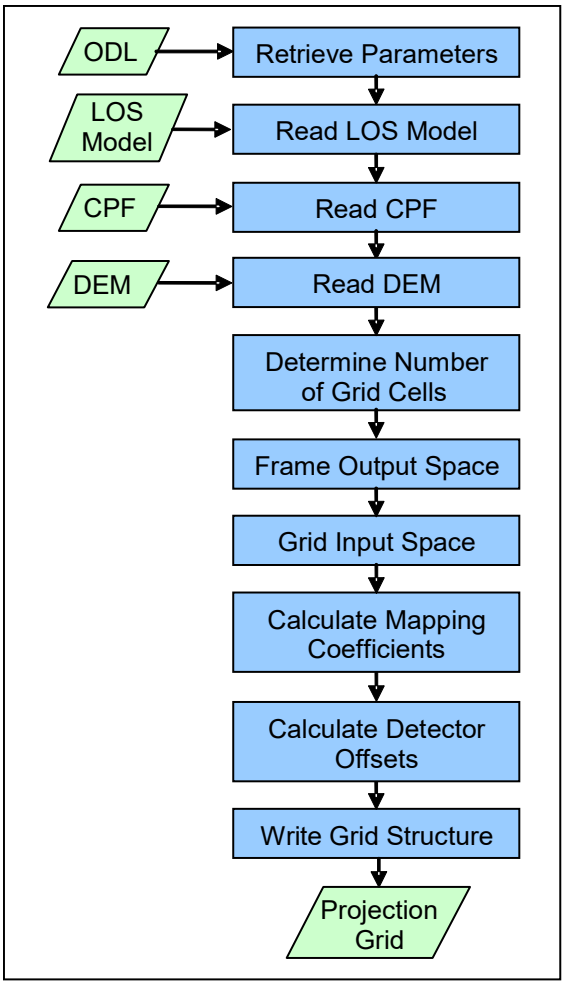

Figure 4-27. Line-of-Sight Projection Block Diagram

4.2.2.6.2.1 Stage 1 - Data Input

The data input stage involves loading the information required to perform grid processing. This includes reading the framing parameters for the output scene from the ODL file, reading grid structural parameters from the CPF, loading the LOS model structure in preparation for invoking the forward model, and reading the DEM to determine the elevation range for the image.

4.2.2.6.2.2 Stage 2 - Scene Framing

Framing the output image space involves determining the geographic extent of the output image to be generated by the resampler. This geographic extent of the output image space is referred to as the output space “frame,” and is specified in output image projection coordinates. Four different methods are used to determine the output frame for Earth-viewing acquisitions. Scene framing for lunar and stellar scenes uses either a maximum bounding rectangle (maxbox) or a minimum bounding rectangle (minbox) approach using inertial LOS declination and right ascension coordinates, and is discussed separately. These methods use the calculated coverage bounds of each

band/SCA in different ways, with some excluding the leading and trailing SCA imagery based on a calculated active image area, and some including the leading/trailing imagery so as to preserve all available input pixels (e.g., for calibration purposes). Thus, the calculation of the active image area for each band is the first step in scene framing.

4.2.2.6.2.2.1 Calculating the Active Image Area

The along-track offsets between spectral bands and even/odd SCAs create an uneven coverage pattern when projected into output image space. In order to provide a more regular output image coverage boundary, we define a rectangular active image area that excludes the excess trailing imagery from even SCAs and the excess leading imagery from odd SCAs. This active area is used for the minbox framing methods, which seek to limit the output product area to provide consistent, contiguous coverage, but are ignored for maxbox framing methods, where all available imagery is desired.

The active image area is computed by constructing 8 critical SCA corner points, labeled C1 through C8 in Figure 4-28. Points C1 and C2 define the top edge of the active area, C3 and C4 the right edge, C5 and C6 the bottom edge, and C7 and C8 the left edge. Note that points C1 and C8 are the same (the upper-left corner of SCA01), as are points C4 and C5 (the lower-right corner of SCA14). The forward model projects these eight line/sample locations to object space, computing the latitude/longitude coordinates of the WGS84 ellipsoid intersection for each point.

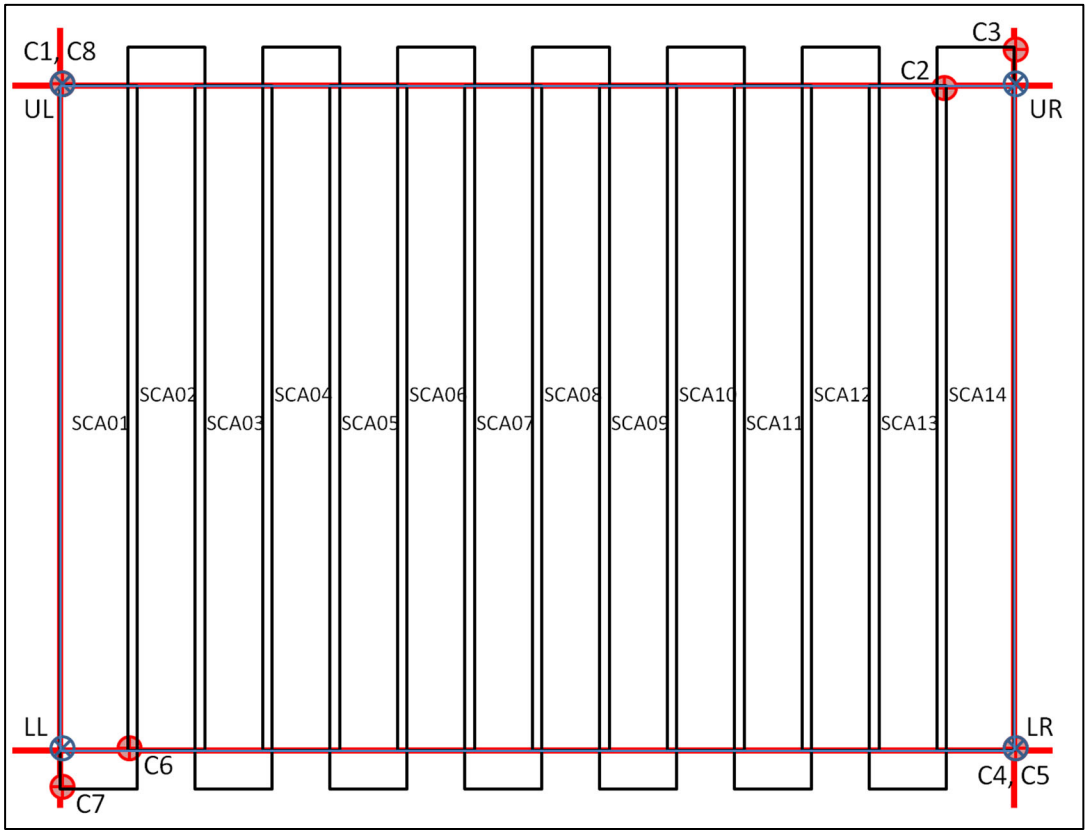

Figure 4-28. Active Image Area Construction

The SCA and corner point assignments are made automatically by examining the SCA across-track and along-track Legendre coefficients to determine: 1) whether SCA01 is on the left (+Y) or right (-Y) side of the scene; 2) whether even or odd SCAs lead; and 3) whether the sample number increases in the –Y or +Y direction. If the across-track Legendre constant term (coef_y0) for SCA01 is positive, then it is the left-most SCA and SCA14 is the right-most. If the along-track Legendre constant term (coef_x0) for SCA01 is greater than that for SCA02, then the odd SCAs lead. If the across-track Legendre linear term (coef_y1) for SCA01 is negative, then the sample number increases in the –Y direction.

Having determined the orientation of the SCAs, we assign the top edge of the active area to the left-most leading SCA UL corner and the right-most leading SCA UR corner, the right edge to the right-most SCA UR and LR corners, the bottom edge to the right-most trailing SCA LR corner and left-most trailing SCA Lower Left (LL) corner, and the left edge to the left-most SCA LL and UL corners. As shown in the figure, for the OLI: C1 = SCA01 (left-most odd SCA) UL, C2 = SCA13 (right-most odd SCA) UR, C3 = SCA14 (right-most SCA) UR, C4 = SCA14 (right-most SCA) LR, C5 = SCA14 (right-most even SCA) LR, C6 = SCA02 (left-most even SCA) LL, C7 = SCA01 (left-most SCA) LL, and C8 = SCA01 (left-most SCA) UL.

The geodetic latitudes computed by the forward model are converted to geocentric latitudes using the following:

$\theta = \arctan((1-e^2) \tan(\phi))$

where: θ = geocentric latitude
ϕ = geodetic latitude
e^2 = WGS84 ellipsoid eccentricity squared

This creates a set of 8 geocentric latitude/longitude (θ_i, λ_i) pairs, one for each “critical” corner, noting that geocentric longitude is equal to geodetic longitude.

Use the geocentric latitude/longitude to construct a geocentric unit vector for each

corner: $$X_i = \begin{bmatrix} \cos(\lambda_i)\cos(\theta_i) \\ \sin(\lambda_i)\cos(\theta_i) \\ \sin(\theta_i) \end{bmatrix}$$

Note that these vectors are inherently normalized.

Construct vectors normal to the top, right, bottom, and left edge great circles by taking cross products of the corner vectors:

$$X_T = \frac{X_1 \times X_2}{|X_1 \times X_2|} \qquad X_R = \frac{X_3 \times X_4}{|X_3 \times X_4|} \qquad X_B = \frac{X_5 \times X_6}{|X_5 \times X_6|} \qquad X_L = \frac{X_7 \times X_8}{|X_7 \times X_8|}$$

Construct corner vectors from the edge vectors:

$$X_{UL} = \frac{X_T \times X_L}{|X_T \times X_L|} \qquad X_{UR} = \frac{X_R \times X_T}{|X_R \times X_T|} \qquad X_{LL} = \frac{X_L \times X_B}{|X_L \times X_B|} \qquad X_{LR} = \frac{X_B \times X_R}{|X_B \times X_R|}$$

The top and bottom edges are next checked against all of the SCA corners to ensure that any curvature in the SCA field angle pattern is accounted for. This is done to suppress residual SCA edge "raggedness."

Adjust the top edge:

Construct a vector in the plane of the top edge great circle:

$$X_g = \frac{(X_{UR} - X_{UL}) \times X_T}{|(X_{UR} - X_{UL}) \times X_T|}$$

Initialize the minimum "out of plane" distance: $a_{min} = 1$

For each SCA:

For the two upper corners: UL (0,0) and UR (ns-1,0):

Use the forward model to project the corner.

Convert the geodetic latitude to geocentric latitude, as above.

Construct a geocentric unit vector, X_i, as above.

Project the unit vector onto the X_g and X_T vectors and compute the ratio:

$$a_i = \frac{X_i \bullet X_T}{X_i \bullet X_g}$$

If $a_i < a_{min}$

$a_{min} = a_i$

$X_{min} = X_i$

Next corner

Next SCA

If $a_{min} < 0$, then the innermost corner lies inside the current active area, and we need to adjust the top edge:

$$X'_g = \frac{(X_{\min} \bullet X_T)X_T + (X_{\min} \bullet X_g)X_g}{|(X_{\min} \bullet X_T)X_T + (X_{\min} \bullet X_g)X_g|}$$

$$X'_T = \frac{X'_g \times (X_{UR} - X_{UL})}{|X'_g \times (X_{UR} - X_{UL})|}$$

Update the top corner vectors using the adjusted edge vectors:

$$X_{UL} = \frac{X'_T \times X_L}{|X'_T \times X_L|} \qquad X_{UR} = \frac{X_R \times X'_T}{|X_R \times X'_T|}$$

Adjust the bottom edge:

Construct a vector in the plane of the bottom edge great circle:

$$X_g = \frac{(X_{LL} - X_{LR}) \times X_B}{\left|(X_{LL} - X_{LR}) \times X_B\right|}$$

Initialize the minimum "out of plane" distance: $a_{min} = 1$

For each SCA:

For the two lower corners: LL (0,nl-1) and LR (ns-1,nl-1):

Use the forward model to project the corner.

Convert the geodetic latitude to geocentric latitude, as above.

Construct a geocentric unit vector, X_i, as above.

Project the unit vector onto the X_g and X_B vectors and compute the ratio:

$$a_i = \frac{X_i \bullet X_B}{X_i \bullet X_g}$$

If $a_i < a_{min}$

$a_{min} = a_i$

$X_{min} = X_i$

Next corner

Next SCA

If $a_{min} < 0$, then the innermost corner lies inside the current active area, and we need to adjust the bottom edge:

$$X'_g = \frac{(X_{\min} \bullet X_B)X_B + (X_{\min} \bullet X_g)X_g}{\left|(X_{\min} \bullet X_B)X_B + (X_{\min} \bullet X_g)X_g\right|}$$

$$X'_B = \frac{X'_g \times (X_{LL} - X_{LR})}{\left|X'_g \times (X_{LL} - X_{LR})\right|}$$

Update the bottom corner vectors using the adjusted edge vectors:

$$X_{LL} = \frac{X_L \times X'_B}{\left|X_L \times X'_B\right|} \qquad X_{LR} = \frac{X'_B \times X_R}{\left|X'_B \times X_R\right|}$$

Convert the four corner vectors to the corresponding geodetic latitude/longitude:

λ = atan2(X.y, X.x)

θ = atan2(X.z, $\sqrt{X.x^2 + X.y^2}$)

ϕ = atan(tan(θ) / (1-e^2))

The four latitude/longitude corners are the bounds of the active image area.

Once the active image area bounds are calculated, the output product frame is determined using one of the following methods:

Method 1: PROJBOX

The user defines the upper-left and lower-right corner coordinates of the area of interest in target map projection coordinates. These coordinates are then projected to the output projection coordinate system using the Projection Transformation Package (see the Projection Transformation sub-algorithm below). This usually results in a non-rectangular area, so a minimum-bounding rectangle is found (in terms of minimum and maximum X and Y projection coordinates) in the resulting output space. This minimum-bounding rectangle defines the output space frame. The output image pixel size is then applied to the projection space to determine the number of lines and samples in the output space. This creates an output image that is map projection north-up.

Method 2: MINBOX

The image active areas for each band, calculated previously, are converted to the specified output map projection coordinate system and used in a minimum bounding rectangle computation to create an output image frame that includes the active area for each band. The computed (latitude/longitude) active area corners are maintained in the grid for subsequent use by the image resampler, so that the output product image will not include leading/trailing SCA imagery.

Method 3: MAXBOX

The four corners of each SCA in each band are projected to the Earth. The maximum and minimum latitude and longitude found across all SCAs and all bands are used to establish the output scene frame in the manner described above for the PROJBOX method. This creates an output frame that contains all input pixels from all bands. The previously calculated image active areas are ignored in this process, and the band active area corners are all set equal to the output product corners. Leading and trailing SCA imagery is thereby not excluded from MAXBOX framed products.

Method 4: PATH

The user specifies a path-oriented Landsat product in either the SOM or UTM projection. In this case, the framing coordinates are not user-specified. The standard path-oriented frame is a preset number of lines and samples based on the Landsat WRS scene size and the maximum rotation needed to create a path-oriented product. Additional options exist to apply either MINBOX or MAXBOX logic in determining the path-oriented product frame.

Method 5: LUNAR

Lunar image framing applies either the same framing methodology as MAXBOX, defining the maximum and minimum corners in right ascension and declination angles with respect to the ECI coordinate system determined by the corners of all the SCAs for all bands, or with a similar framing methodology as MINBOX, determining the corners based solely on the SCA that contains the moon. The

right ascension and declination angles are adjusted according to the change in orbit of the moon during image acquisition.

Method 6: STELLAR

Stellar image framing applies the same framing methodology as MAXBOX, only the output space frame defining the maximum and minimum corners are in right ascension, and declination angles with respect to the ECI coordinate system.

The scene framing logic uses the following sub-algorithms/routines:

a) Validate UTM Zone

The nominal UTM zone to use is computed from the scene center longitude, but the projection may be forced to an adjacent zone using input parameters. In particular, each WRS path/row may be preassigned to a UTM zone so that the same zone is always used for scenes near UTM zone boundaries. This should not introduce a zone offset greater than 1. The validation is performed by computing the UTM zone in which the scene center falls and then determining whether the input UTM zone (if any) is within one zone of the nominal zone.

Shift the scene center longitude to put it in the range 0-360 degrees:
SC_long = mod(SC_long + 540, 360)
where: SC_long is the scene center longitude in degrees

Compute the nominal UTM zone (note that UTM zones are six degrees wide):
SC_zone = (int)floor(SC_long/6) + 1

See if the input zone is within one zone of the nominal zone:
if (abs(input_zone - SC_zone) < 2 or (60 - abs(input_zone - SC_zone)) < 2)
then input_zone is valid.

b) North Up Framing

Determine the scene corners. Scene corners depend on whether the corners were user input (PROJBOX) or calculated by projecting the Level 1R image corners (MAXBOX), but the framing logic is essentially the same in each case. Once given as input or computed, the latitude/longitude scene corners are converted to the defined map projection, the extreme X and Y coordinates are found, and these extreme points are rounded to a whole multiple of the pixel size. The following sub-algorithms describe each of the north-up framing methods.

b).1. Map Edge/PROJBOX Framing

Calculates the minimum and maximum projection coordinates for given upper-left and lower-right latitude, longitude coordinates.

- Calculate the min/max coordinates along the east edge of the output area by computing latitude/longitude to map x/y projections for a series of points from

(minimum latitude, maximum longitude) to (maximum latitude, maximum longitude).
- Calculate the min/max coordinates along the west edge of the output area by computing latitude/longitude to map x/y projections for a series of points from (minimum latitude, minimum longitude) to (maximum latitude, minimum longitude).
- Calculate the min/max coordinates along the south edge of the output area by computing latitude/longitude to map x/y projections for a series of points from (minimum latitude, minimum longitude) to (minimum latitude, maximum longitude).
- Calculate the min/max coordinates along the north edge of the output area by computing latitude/longitude to map x/y projections for a series of points from (maximum latitude, minimum longitude) to (maximum latitude, maximum longitude).

Note that since lines of constant latitude and/or longitude may be curved in map projection space, the extreme map x/y points may not correspond to the four PROJBOX corners.

b).2. Minbox/Maxbox Framing Determine the frame in output space for the minbox or maxbox north-up product. The actual frame is determined based on the optimal band's pixel size, but the frame is the same for every band.

b).2.1 Minbox Framing Calculate the MINBOX frame bounds, using the active area corner points for each band.

1. Call projtran (see below) to get the output map projected x/y for each active area corner point for each image band.
2. Find the minimum and maximum output proj x/y from the full set of active area corner points.
3. Pad the min and max output projection x/y to make them a multiple of pixsize.
4. Fill in the corners for the grid in the order of UL, LL, UR, LR, and Y/X coordinates.
   UL = min x, max y
   UR = max x, max y
   LL = min x, min y
   LR = max x, min y
5. Find the number of lines and samples for the grid, for each specified band number.
   lines = (max y - min y)/pixsize + 1
   samples = (max x - min x)/pixsize + 1

b).2.2 Maxbox Framing Calculate the MAXBOX product frame bounds, using the projected corners of each band/SCA.

1. Find the four image corners in input space for each SCA and band.

UL - (1, first_pixel)
UR - (1, last_pixel)
LL - (NLines, first_pixel)
LR - (NLines, last_pixel)

2. Call the forward model (see below) to get the output lat/long, for each corner point.
3. Call projtran (see below) to get the output map projected x/y, for each corner point.
4. Find the minimum and maximum output proj x/y from the full set of corner points.
5. Pad the min and max output projection x/y to make them a multiple of pixsize.
6. Fill in the corners for the grid in the order of UL, LL, UR, LR and Y/X coordinates.
   UL = min x, max y
   UR = max x, max y
   LL = min x, min y
   LR = max x, min y
7. Find the number of lines and samples for the grid, for each specified band number.
   lines = (max y - min y)/pixsize + 1
   samples = (max x - min x)/pixsize + 1
8. Call projtran to convert the map projection Y/X coordinates of the output product corners to latitude/longitude.
9. Replace the active area corner coordinates for each band with the converted output product corner coordinates.

b).2.3. Pad Corners Pad the input corners by a defined factor of the pixel size. The x/y min and max values are input for the corner locations. These values are padded by PADVAL * the pixel size. The newly padded x/y min and max values are returned, replacing the original values.

ixmin = int (Xmin/(PADVAL*pixsize))
Xmin = ixmin*PADVAL*pixsize
ixmax = int (Xmax/(PADVAL*pixsize))+1
Xmax = ixmax*PADVAL*pixsize
iymin = int (Ymin/(PADVAL*pixsize))
Ymin = iymin*PADVAL*pixsize
iymax = int (Ymax/(PADVAL*pixsize))+1
Ymax = iymax*PADVAL*pixsize

c) Path-oriented Framing
Provide a path-oriented projection that is framed to a nominal WRS scene. The projection, pixel size, and the path and row of the scene must be defined.

c).1. Calculate Center and Rotation Angle
Calculate the scene center and rotation angle for a nominal WRS scene. The WRS path and row of the input scene and the projection parameters are needed as input. The nominal WRS scene center lat/long and rotation angle for the given projection are returned. The algorithm has the following steps:

Convert input angles to radians:
Inclination_Angle_R = Pi / 180 * Inclination_Angle
Long_Path1_Row60_R = Pi / 180 * Long_Path1_Row60

Compute the Earth's angular rotation rate:
earth_spin_rate = 2 * Pi / (24 * 3600)

Note: We use the solar rotation rate rather than the sidereal rate in order to account for the orbital precession, which is designed to make the orbit sun synchronous. Thus, the apparent Earth angular velocity is the inertial (sidereal) angular velocity plus the precession rate, which, by design, is equal to the solar angular rate.

Compute the spacecraft's angular rotation rate:
SC_Ang_Rate = 2 * Pi * WRS_Cycle_Orbits / (WRS_Cycle_Days*24*3600)

Compute the central travel angle from the descending node:
Central_Angle = (Row - Descending_Node_Row)/Scenes_Per_Orbit*2*Pi

Compute the WRS geocentric latitude:
WRS_GCLat = asin(-sin(Central_Angle) * sin(Inclination_Angle_R))

Compute the longitude of Row 60 for this Path:
Long_Origin = Long_Path1_Row60_R - (Path-1) * 2*Pi/WRS_Cycle_Orbits

Compute the WRS longitude:
Delta_Long = atan2(tan(WRS_GCLat)/tan(Inclination_Angle_R),
cos(Central_Angle)/cos(WRS_GCLat))
WRS_Long = Long_Origin - Delta_Long - Central_Angle *
Earth_Spin_Rate / SC_Ang_Rate

Make sure the longitude is in the range +/- Pi:
While (WRS_Long > Pi) WRS_Long = WRS_Long - 2*Pi
While (WRS_Long < -Pi) WRS_Long = WRS_Long + 2*Pi

Compute the scene heading:
Heading_Angle = atan2(cos(Inclination_Angle_R)/cos(WRS_GCLat),
-cos(Delta_Long)*sin(Inclination_Angle_R))

Convert the WRS geocentric latitude to geodetic latitude:
WRS_Lat = atan(tan(WRS_GCLat) * (Semi_Major_Axis/Semi_Minor_Axis) *
(Semi_Major_Axis/Semi_Minor_Axis))

Convert angles to degrees:
WRS_Lat = WRS_Lat * 180 / Pi
WRS_Long = WRS_Long * 180 / Pi

Heading_Angle = Heading_Angle * 180 / Pi

Round WRS lat/long off to the nearest whole arc minute:
WRS_Lat = round(WRS_Lat*60) / 60
WRS_Long = round(WRS_Long*60) / 60

c).2. Calculate Path-Oriented Frame
Calculate the center point and rotation angle, and the image corner coordinates in a SOM or UTM projection. Also calculate the first-order polynomial coefficients, which map output line/sample coordinates to their corresponding output projection coordinates. Determine the frame in output space for the path-oriented product. Calculate the frame for each band. The frame must be the same for all bands.

c).2.1. Angle to Map
Convert the WRS rotation angle (from geodetic north) to a frame orientation angle in map coordinates. The following is an algorithm to compute this:

Convert the WRS scene center latitude/longitude to map projection x/y (X1, Y1) using the projtran routine.

Add 1 microradian (0.2 seconds) to the WRS scene center latitude and convert this point to map projection x/y (X2, Y2).

Compute the azimuth of this line in grid space as the arctangent of (X2-X1)/(Y2-Y1). This is the grid azimuth of geodetic north at the WRS scene center.

Add this angle to the WRS rotation angle to give the grid heading. A standard framed scene puts the satellite direction of flight at the bottom of the scene, so the scene orientation angle is the grid heading + or - 180 degrees. If the grid heading is <0, then subtract 180 degrees. If the grid heading is >0, then add 180 degrees. This is the scene orientation angle to use with the WRS scene center.

c).2.2. Path-oriented Minbox/Maxbox Frame
Calculate the path-oriented frame that is large enough to contain all bands.

c).2.2.1. Calculate the Path-oriented Minbox Frame
Calculate the path-oriented frame for the minbox approach.

1. Compute the map projection coordinates of the four image active area corners for each band, as described in step 1 of Minbox Framing.
2. Offset and rotate the scene corners to the path-oriented frame using the WRS scene center map projection coordinates (X1, Y1) and orientation angle:
    a. X' = (X - X1) cos(angle) - (Y - Y1) sin(angle) + X1
    b. Y' = (X - X1) sin(angle) + (Y - Y1) cos(angle) + Y1
3. Compute the minbox frame as described in steps 2-4 of Minbox Framing.

4. Convert the rotated minbox corners back to the unrotated map projection coordinate system:
    a. X = (X' - X1) cos(angle) + (Y' - Y1) sin(angle) + X1
    b. Y = -(X' - X1) sin(angle) + (Y' - Y1) cos(angle) + Y1

c).2.2.2. Calculate Path-oriented Maxbox Frame

Calculate the path-oriented frame for the maxbox approach.

1. Compute the map projection coordinates of the four image corners for the optimal band, as described in steps 1-3 of Maxbox Framing.
2. Offset and rotate the scene corners to the path-oriented frame, using the WRS scene center map projection coordinates (X1, Y1) and orientation angle:
    a. X' = (X - X1) cos(angle) - (Y - Y1) sin(angle) + X1
    b. Y' = (X - X1) sin(angle) + (Y - Y1) cos(angle) + Y1
3. Compute the maxbox frame, as described in steps 4-6 of Maxbox Framing.
4. Convert the rotated maxbox corners back to the unrotated map projection coordinate system:
    a. X = (X' - X1) cos(angle) + (Y' - Y1) sin(angle) + X1
    b. Y = -(X' - X1) sin(angle) + (Y' - Y1) cos(angle) + Y1
5. Call projtran to convert the map projection Y/X coordinates of the output product corners to latitude/longitude.
6. Replace the active area corner coordinates for each band with the converted output product corner coordinates.

d) Celestial Acquisitions

Celestial acquisitions use the same framing logic as Earth acquisitions (namely maxbox), but the output space coordinate systems are sufficiently different to merit separate discussion. For both lunar and stellar acquisitions, the output space is defined in terms of directions in inertial space, defined by the ECI J2000 right ascension and declination of the OLI look vectors. In the case of stellar acquisitions, the output space "projection" uses the ECI J2000 right ascension and declination directly. For lunar acquisitions, the output coordinate system is modified to use the LOS right ascension and declination offset from the lunar right ascension and declination at the time of observation. This creates a slowly rotating coordinate system that tracks the moon and is the reason for having a planetary ephemeris file as an input to this algorithm. These differences emerge in the forward model computations for celestial acquisitions, where the LOS intersection logic used for Earth acquisitions is replaced by operations on the inertial lines-of-sight (after conversion to inertial right ascension and declination angles), with the resulting map projection x/y coordinates used in the Earth-view algorithms replaced by right ascension and declination (or delta-right ascension and delta-declination). Either the maxbox or minbox framing logic applied to the x/y map projection coordinates in Earth-view acquisitions is then applied to these angular celestial coordinates.

e) Lunar Acquisitions

Lunar acquisitions use either a MAXBOX or MINBOX framing type. For MAXBOX, the framing logic is the same as that used for Earth viewing acquisitions; determine bounding viewing angles based on all SCAs of all bands. For MINBOX, the minimum box, or viewing angles, is based on the SCA within which the moon resides. The bounding viewing angles for this SCA for all bands define the frame for the output image. The moon is found to reside within an SCA by using the moon's coordinates, defined by the center of acquisition time of the scene, and checking to see if these coordinates fall within a SCA. A simple point in a polygon routine is used for the check:

If coordinate to check is defined as X_m and Y_m
e1) Define rectangle using SCA corners, X_i and Y_i.
e2) Find maximum Y coordinate of rectangle, Y_{max}.
e3) Define a new coordinate as the following:
$X_n = X_m$
Y_n = Delta + Y_{max}
Where Delta is large enough to put point (X_n, Y_n) outside of polygon
e4) Define a line from (X_m,Y_m) to (X_n,Y_n).
e5) Determine the number of times the line defined in e4) intersects sides of the rectangle from e1). If the number of intersections is an odd number, then the point is within the rectangle.

4.2.2.6.2.3 Stage 3 - Grid Definition

The grid definition stage determines the required size of the grid, allocates the grid structure, and computes the input space (Level 1R) line/sample locations for each grid cell.

a) Determine the Number of Grid Input/Output Lines/Samples
Determine the number of input points to be stored in the grid according to the grid-sampling rate or grid cell size chosen.

Loop through each band stored in the grid.

Loop through each SCA stored in the grid.

Calculate the number of lines and samples stored in the grid, according to the size of each grid cell and the size of the input image to be processed. Store the number of grid lines and samples calculated in the grid.

Calculate the number of times grid cell size divides into Level 1R imagery

$$\text{number of grid lines} = \frac{\text{number of image lines}}{\text{grid cell size line direction}} + 1$$

$$\text{number grid samples} = \frac{\text{number of detectors per SCA}}{\text{grid cell size sample direction}} + 1$$

where:

number of image lines = number of lines in Level 1R (LOS model)
number of detectors per SCA = number of samples per SCA (LOS model)
grid cell size line direction = number of lines in one grid cell
grid cell size sample direction = number of samples in one grid cell

If the grid cell size in the line direction does not divide evenly into the number of lines in the Level 1R, then increment the number of grid lines by one.

If the grid cell size in the sample direction does not divide evenly into the number of samples in the Level 1R, then increment the number of grid samples by one.

b) Determine Grid Lines/Samples

Determine where each grid cell point will fall in the input Level 1R image. These grid cell points will fall at integer locations in the input imagery.

Loop through each band that is stored in the grid.

Loop through each SCA stored in the grid.

Initialize the first grid cell line location to zero relative.

input line location grid $cell_0 = 0$

Loop until the grid cell line location is greater than or equal to the number of Level 1R lines, incrementing each new grid cell line location by the appropriate grid cell size in the line direction for the current band and SCA.

input line location grid $cell_n$ = input line location grid $cell_{n-1}$ + grid cell size line direction

Set the last grid cell line location to the last line in Level 1R image.

input line location grid $cell_{last}$ = number of lines in Level 1R imagery

Initialize the first grid cell sample location to zero relative.

input sample location grid $cell_0 = 0$

Loop until the grid cell sample location is greater than or equal to the number of Level 1R samples, incrementing each new grid cell sample location by the appropriate grid cell size in the sample direction for the current band and SCA.

input sample location grid $cell_n$ = input sample location grid $cell_{n-1}$ + grid cell size sample direction

Set the last grid cell sample location to the last sample in Level 1R image.
input sample location grid $cell_{last}$ = number of samples in Level 1R imagery

4.2.2.6.2.4 Stage 4 - Grid Construction

Once the grid structures are created (one per SCA per band), the forward model is evaluated at every grid intersection; that is, for every Level 1R line/sample location at every elevation plane. The forward model computes the WGS84 latitude/longitude coordinates associated with each input line/sample/height point. These latitude/longitude positions are then converted to output space line/sample by projecting them to map x/y, computing the offsets (and rotation if path-oriented) from the upper-left scene corner, and scaling the offsets from meters to pixels using the pixel size.

a) Make Grid

Given the number of grid lines and samples that will be sampled in the input imagery, loop on each band of each SCA, loop on the number of z-planes, loop on the number of input grid lines and samples calculating the corresponding output line and sample location. For each input line, sample location, and elevation, the instrument forward model function is called. The steps below outline this forward model function. Additional detail on the sub-algorithms, which comprise the forward model, is provided in the subsection titled "Forward Model" later in this document.

The forward model uses the LOS model structure and the CPF to map an input line and sample location to an output geographic location. These are the steps performed whenever calculating an output geodetic latitude and longitude from an input line and sample by invoking the instrument “forward model.” The GCTP function can then be used to transform the geographic latitude and longitude to a map projection X and Y coordinate. If the output image has a “North up” orientation, then the upper-left projection coordinate of the output imagery and the output pixel size can be used to transform any projection coordinate to an output line and sample location. If the map projection space is in a rotated projection space, such as having a satellite path orientation, then a transformation handling rotation is established between the projection space and output pixel location. This transformation is then used in converting projection coordinates to output pixel line and sample locations.

The process listed below is performed on all bands, all elevation planes, and all SCAs present in the grid. The detector type used in the process is nominal (see the LOS Model Creation Algroithm (4.2.1) for a discussion of detector types). The list explains the actions taken if a detector type other than nominal is chosen, so that it can be referenced later.

Loop on the number of input grid lines.
Loop on the number of input grid samples.

Read the input space (Level 1R) line/sample coordinate for this grid point.

Loop on the number of elevation planes.

Compute the height of the current elevation plane:

height = $(z - z_{elev=0})$ * (elevation increment)

where:

z is the index of the current z-plane and
$z_{elev=0}$ is the index of the zero elevation z-plane.

Invoke the forward model to compute the corresponding ground position latitude/longitude for this point. The general steps of the forward model are described here and are presented in more detail below.

Find Time
Find the nominal time of the input sample relative to the start of the imagery. The LOS Model Creation Algorithm (4.2.1) describes this procedure, and it is presented below in the Find Time sub-algorithm description.

Find LOS
Find the LOS vector for the input line/sample location using the Legendre polynomial coefficients, as described below in the Find LOS sub-algorithm.

Find Attitude
Calculate the spacecraft attitude corresponding to the LOS, i.e., for the line/sample location, at the time computed above, using the Find Attitude sub-algorithm described below. Note that for Earth acquisitions, the roll-pitch-yaw attitude sequence in the LOS model is relative to the orbital coordinate system, whereas for celestial (lunar/stellar) acquisitions, the LOS model roll-pitch-yaw sequence is with respect to the ECI J2000 coordinate system. The operations applied by the Find Attitude sub-algorithm are the same in either case.

Find Ephemeris
Calculate the satellite position for the line/sample using Lagrange interpolation. Reference the move_sat sub-algorithm described below. Note that for Earth acquisitions, the move_sat sub-algorithm is provided with the corrected ECEF ephemeris data from the LOS model, whereas for celestial (lunar/stellar) acquisitions, it will be passed the corrected ECI ephemeris.

Rotate LOS to ECEF (Earth-view) or ECI (Celestial)
Use the OLI alignment matrix in the LOS model to convert the LOS vector from sensor to ACS/body coordinates. Then, apply the interpolated roll, pitch, and yaw to the LOS to convert ACS/body to orbital (Earth-view) or ECI (celestial). If Earth-view, use the ephemeris to construct the orbital to ECEF rotation matrix and use it to transform LOS to ECEF. The Attitude sub-algorithm below describes the procedure for Earth-view scenes. For celestial

acquisitions, the procedure is complete once the LOS has been rotated to ECI using the roll-pitch-yaw perturbation matrix.

Spacecraft Center of Mass to OLI Offset Correction
Adjust the spacecraft position for the offset between the spacecraft center of mass and the OLI instrument. This offset, in spacecraft body coordinates, is stored in the LOS model structure. First, convert the offset from spacecraft body frame to ECEF using the attitude perturbation matrix (body to orbital) and the orbital to ECEF matrix:

[orbital CM to OLI] = [perturbation] [body CM to OLI]

[ECEF CM to OLI] = [ORB2ECEF] [orbital CM to OLI]

Add the offset to the ECEF spacecraft position vector. This correction is not used for celestial (lunar/stellar) acquisitions.

Correct LOS for Velocity Aberration
The relativistic velocity aberration correction adjusts the computed LOS (ECEF for Earth-view and ECI for celestial) for the apparent deflection caused by the relative velocity of the platform (spacecraft) and target. The preparatory computations are somewhat different for Earth-view and celestial acquisitions due to the differences in target velocity.

Earth-view Case
The LOS intersection sub-algorithm described below (see Find Target Position) is invoked with an elevation of zero to find the approximate ground target position. The ground point velocity is then computed as:

$\mathbf{V}_g = \omega \times \mathbf{X}_g$

where:
$\mathbf{V}_g$ = ground point velocity
$\mathbf{X}_g$ = ground point ECEF position
$\boldsymbol{\omega}$ = Earth rotation vector = $[\ 0 \quad 0 \quad \Omega_e\ ]^T$
Ω_e = Earth rotation rate in radians/second (from CPF)

The relative velocity is then:

$\mathbf{V} = \mathbf{V}_s - \mathbf{V}_g$

where Vs is the spacecraft ECEF velocity from the ephemeris data.

Correcting the Earth-View LOS
The LOS vector is adjusted based on the ratio of the relative velocity vector to the speed of light (from the CPF):

$$\boldsymbol{l}' = \frac{\boldsymbol{l} - \frac{\boldsymbol{V}}{c}}{\left| \boldsymbol{l} - \frac{\boldsymbol{V}}{c} \right|}$$ where: **l** = uncorrected LOS and **l'** = corrected LOS

Note: In this case, the LOS velocity aberration correction is negative since we are correcting the apparent LOS to the true (aberration corrected) LOS. The correction is positive if we are computing the apparent LOS from the true (geometrical) LOS (see lunar case below).

Celestial (Lunar/Stellar) Case
Both lunar and stellar acquisitions use the spacecraft inertial velocity from the ephemeris data as the relative velocity. This is justified by the use of a lunar ephemeris (using the Naval Observatory's NOVAS-C package) that returns apparent places. The apparent location of the moon is already corrected for light travel time (see below) and velocity/planetary aberration due to the motion of the moon around the Earth. Thus, the residual aberration is due only to the motion of the spacecraft relative to the Earth. Thus, for both lunar and stellar acquisitions:

$\mathbf{V} = \mathbf{V}_s$

where Vs is the spacecraft ECI velocity from the ephemeris data.

Correcting the Celestial LOS
For stellar acquisitions, the LOS is corrected for aberration in the same manner as for Earth-view scenes. For lunar acquisitions, rather than correct the LOS vector, we adjust the apparent location of the moon. The lunar vector is thus adjusted based on the ratio of the relative velocity vector to the speed of light (from the CPF) as the following:

$$\mathbf{l}' = \frac{\mathbf{l} + \frac{\mathbf{V}}{c}}{\left| \mathbf{l} + \frac{\mathbf{V}}{c} \right|}$$ where: **l** = uncorrected LOS and **l'** = corrected LOS

The correction is positive in this case since we are computing an apparent location rather than correcting one.

LOS Intersection
For Earth-view acquisitions, intersect the LOS in ECEF with the Earth model as described in the Find Target Position sub-algorithm below. This yields the geodetic latitude, longitude, and height of the ground point.

For celestial acquisitions, convert the ECI LOS to Right Ascension (RA) and declination (δ) angles:

$$RA = \tan^{-1}(\frac{y}{x})$$

$$\delta = \tan^{-1}(\frac{z}{\sqrt{x^2 + y^2}})$$

where the ECI LOS vector is [x y z]^T.

Correct Ground Position for Light Travel Time

Since the light departing the ground point takes a finite time to arrive at the OLI sensor, there is a slight discrepancy in the corresponding time at the ground point and at the spacecraft. Since the LOS intersection logic assumed that these times were the same, a small correction can be made to correct for this light travel time delay.

Given the ECEF positions of the ground point and the spacecraft, compute the light travel time correction as follows:

Compute the distance from the ground point to the spacecraft:

$$d = \left| \mathbf{X_s} - \mathbf{X_g} \right|$$

where:

X$_s$ is the spacecraft ECEF position and
X$_g$ is the ground point ECEF position.

Compute the light travel time using the speed of light (from CPF):

$$ltt = \frac{d}{c}$$

Compute the Earth rotation during light travel:

$\theta = ltt * \Omega_e$ where Ω_e is the Earth angular velocity from the CPF.

Apply the light travel time Earth rotation:

$$\mathbf{X}'_g = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{X}_g$$

where:

X$_g$**'** is the corrected ECEF position
X_g is the uncorrected ECEF position

Convert the corrected ECEF position to geodetic latitude, longitude, and height.

Note that the light travel time correction for lunar observations due to the difference between the Earth-moon distance and the spacecraft-moon distance is neglected. This is justified by the fact that that the lunar angular rate is less than 3 microradians per second and the maximum LTT difference is about 25 milliseconds, making the magnitude of this effect less than 0.1 microradians.

Convert Position to Output Space Line/Sample
The angular geodetic (latitude/longitude) or celestial (RA/declination) coordinates must be converted to the corresponding output space line/sample coordinate to complete the input space to output space mapping.

For Earth-view acquisitions, this is accomplished as follows:

Calculate the map projection X/Y for the geodetic latitude and longitude.

Convert the map X/Y coordinate to the output line/sample location:

If the output map projection is of a path-oriented projection, then the X/Y coordinate is transformed to output space with a bilinear transformation.

$$line = a_0 + a_1 * X + a_2 * Y + a_3 * X * Y$$
$$sample = b_0 + b_1 * X + b_2 * Y + b_3 * X * Y$$

where:
a_i = polynomial coefficients that map X/Y to an output line location
b_i = polynomial coefficients that map X/Y to an output sample location
X,Y = map projection coordinates

The polynomial transformation is set up to handle the rotation involved in rotating a “Map North” projection to Satellite of “Path” projection (i.e., one that has the output line coordinate system more closely aligned with the along flight path of the satellite).

If the output map projection is not path-oriented, but “North up,” the relationship between X/Y and output line/sample does not involve any rotation, and the following equation is used:

$$\text{line} = \frac{\text{upper left Y} - \text{Y}}{\text{pixel size Y}}$$
$$\text{sample} = \frac{\text{X} - \text{upper left X}}{\text{pixel size X}}$$

where:
upper left Y = the upper-left Y projection coordinate of the output image

upper left X = the upper-left X projection coordinate of the output image
pixel size Y = the output pixel size in Y coordinates
pixel size X = the output pixel size in X coordinates

Note that these line and sample pixel coordinates are (0,0) relative (i.e., the center of the upper-left pixel is at line,sample 0,0).

For lunar acquisitions, the right ascension and declination angles derived from the inertial LOS are offset from the nominal lunar inertial position to establish an output frame that "tracks" the apparent location of the moon. This is done as follows:

a) Compute the apparent ECI J2000 position of the moon.

1. Use the input JPL lunar ephemeris data in the NOVAS-C package to compute the ECITOD apparent location of the moon at the time corresponding to the current LOS (lxx_moonpos). This apparent location is provided as an ECITOD vector (i.e., including both direction and distance).
2. Apply the nutation and precession corrections (see the Ancillary Data Preprocessing Algorithm (4.1.4) for additional information) to convert the ECITOD vector to ECI J2000.
3. Subtract the current spacecraft ECI J2000 position vector from the lunar ECI J2000 vector to compute the spacecraft-lunar vector.
4. Compute the apparent (parallax corrected) right ascension, declination, and spacecraft-lunar distance from the spacecraft-lunar vector (by invoking exx_cart2sph.

b) Compute the differences between the LOS right ascension and declination and the apparent lunar right ascension and declination.

c) Normalize the nominal angular pixel size by the ratio of the current spacecraft-moon distance (computed above) and the nominal spacecraft-moon distance. The nominal distance is computed at the acquisition center time.

$$psize_{current} = psize_{nominal} * distance_{nominal} / distance_{current}$$

d) Divide the angular distances computed in b) above by the normalized pixel size computed in c) above. This yields the moon-relative line/sample coordinate. This is the coordinate space in which lunar images are framed, so the offset between these coordinates and the lunar scene upper-left corner coordinates yields the output space line/sample for the current grid point.

For stellar acquisitions, the right ascension and declination angles derived from the inertial LOS are used directly. The offsets relative to the scene upper-left

corner (in right ascension/declination space) are computed and divided by the angular pixel size to compute output space line/sample coordinates.

One additional note regarding the celestial acquisition scene framing is in order. Since right ascension, like longitude, increases eastward, and declination, like latitude, increases northward, and given that celestial images are looking "up" rather than "down," the right ascension-x, declination-y coordinate system is left-handed. This can lead to the moon being apparently inverted left-to-right in the output image. This is not important for the applications (e.g., band registration characterization) in which the lunar images are to be used. If "anatomically correct" lunar images are required, some changes to the framing logic may be necessary.

The line and sample location calculated is stored in the grid structure. This line/sample location is then the output location for the corresponding input line/sample and the current elevation (current grid line/sample input locations).

b) Calculate Jitter Sensitivity Coefficients

The forward model is invoked multiple times at each grid intersection to compute the effect that small attitude perturbations about each spacecraft axis have on the input space to output space line/sample mapping. This is done at each grid point as follows:

Save the current grid point input line/sample as in_line/in_samp and the current grid point output line/sample as line0/samp0.

For each spacecraft axis (roll-pitch-yaw) :

1. Perturb the attitude about the selected axis by 1 microradian.
2. Use the forward model to compute the output line/sample corresponding to the current input line/sample using the perturbed attitude, and store the result in line[0]/samp[0].
3. Perturb the input line number by 1 line (delta_line = 1) and recompute the corresponding output line/sample, storing the result in line[1]/samp[1].
4. Restore the input line number to in_line and perturb the input sample number by 1 sample (delta_samp = 1), and recompute the corresponding output line/sample, storing the result in line[2]/samp[2].
5. Calculate the output space to input space line/sample sensitivities as follows:
   a. delta_oline_per_iline = (line[1]–line[0]) / delta_line
   b. delta_oline_per_isamp = (line[2]–line[0]) / delta_samp
   c. delta_osamp_per_iline = (samp[1]–samp[0]) / delta_line
   d. delta_osamp_per_isamp = (samp[2]–samp[0]) / delta_samp
6. Invert the resulting 2-by-2 sensitivity matrix to find the input line/samp per output line/samp sensitivities:
   a. determinant = delta_oline_per_iline * delta_osamp_per_isamp – delta_oline_per_isamp * delta_osamp_per_iline
   b. delta_iline_per_oline = delta_osamp_per_isamp / determinant
   c. delta_iline_per_osamp = -delta_oline_per_isamp / determinant
   d. delta_isamp_per_oline = -delta_osamp_per_iline / determinant
   e. delta_isamp_per_osamp = delta_oline_per_iline / determinant

7. Apply the input line/samp per output line/samp sensitivities to the output line/samp offset due to the attitude perturbation, to find the equivalent input space offset :
   a. d_iline = delta_iline_per_oline * (line[0] – line0) + delta_iline_per_osamp * (samp[0] – samp0)
   b. d_isamp = delta_isamp_per_oline * (line[0] – line0) + delta_isamp_per_osamp * (samp[0] – samp0)
8. Divide by the attitude perturbation to compute the input line/sample to attitude jitter sensitivities for this axis at this grid point:
   a. line_sens[axis] = -d_iline / perturbation
   b. samp_sens[axis] = -d_isamp / perturbation

   Where :

   line_sens[] is the array of roll-pitch-yaw line sensitivities for the grid.
   samp_sens[] is the array of roll-pitch-yaw sample sensitivities for the grid.
   perturbation is the 1 microradian attitude perturbation introduced in step 1.

   Note that the sign of the sensitivities is inverted in this calculation. This is done because the sensitivities will be used to compute the equivalent input space corrections needed to compensate for an attitude disturbance. Therefore, since d_iline is the input space line offset that is equivalent to one microradian of jitter for the current axis, an offset of –d_iline will compensate for this jitter.

A 2-by-3 array containing the line and sample sensitivity coefficients for the roll, pitch, and yaw axes is stored for each grid point.

c) Calculate Map Coefficients

Bilinear mapping coefficients for each grid cell are calculated for mapping from the input location to output location (forward mapping) and for mapping from the output location to input location (inverse mapping). A separate mapping function is used for lines and samples. This equates to four mapping functions. A set of four mapping functions is calculated for each grid cell, for each SCA, for every band, and for every elevation plane that is stored in the grid.

The following methodology is used for calculating one set of four bilinear mapping equations:

A 9x4 matrix is used to fit nine points within a grid cell. The matrix equation takes the form of:

$$[A][coeff] = [b]$$

In this equation, matrix A is 9x4, vector b is 9x1, and the coefficient matrix is 4x1. The coefficient matrix, [*coeff*], can be solved to obtain the mapping coefficients as follows:

$$[coeff] = [A^T A]^{-1}[A^T b]$$

In solving for an equation to map an input line and sample location to an output sample location, belonging to one grid cell, the matrices can be defined as follows:

$A_{n,0} = 1$ where n=0,8

$A_{0,1}$ = upper-left input sample location for the current grid cell
$A_{1,1}$ = upper-right input sample location for the current grid cell
$A_{2,1}$ = lower-left input sample location for the current grid cell
$A_{3,1}$ = lower-right input sample location for the current grid cell
$A_{4,1} = (A_{0,1}+A_{1,1}+A_{2,1}+A_{3,1})/4$
$A_{5,1} = (A_{0,1}+A_{1,1})/2$
$A_{6,1} = (A_{1,1}+A_{3,1})/2$
$A_{7,1} = (A_{2,1}+A_{3,1})/2$
$A_{8,1} = (A_{2,1}+A_{0,1})/2$
$A_{0,2}$ = upper-left input line location for the current grid cell
$A_{1,2}$ = upper-right input line location for the current grid cell
$A_{2,2}$ = lower-left input line location for the current grid cell
$A_{3,2}$ = lower-right input line location for the current grid cell
$A_{4,2} = (A_{0,2}+A_{1,2}+A_{2,2}+A_{3,2})/4$
$A_{5,2} = (A_{0,2}+A_{1,2})/2$
$A_{6,2} = (A_{1,2}+A_{3,2})/2$
$A_{7,2} = (A_{2,2}+A_{3,2})/2$
$A_{8,2} = (A_{2,2}+A_{0,2})/2$
$A_{n,3} = A_{n,1}*A_{n,2}$ where n=0…8

b_0 = upper-left output sample location for the current grid cell
b_1 = upper-right output sample location for the current grid cell
b_2 = lower-left output sample location for the current grid cell
b_3 = lower-right output sample location for the current grid cell
$b_4 = (b_0+b_1+b_2+b_3)/4$
$b_5 = (b_0+b_1)/2$
$b_6 = (b_1+b_3)/2$
$b_7 = (b_2+b_3)/2$
$b_8 = (b_2+b_0)/2$

The line and sample locations listed above are defined at the grid cell corners coordinates. The points interpolated in between the grid cell line segments provide stability for what could be, for a mapping that involves a 45° rotation, an ill-defined solution if only four points were used in the calculation. The set of coefficients define a bilinear mapping equation of the form:

$$sample_o = coeff_0 + coeff_1 * sample_i + coeff_2 * line_i + coeff_3 * sample_i * line_i$$

where:

$sample_o$ = output sample location
$sample_i$ = input sample location
$line_i$ = input line location

The forward mapping equations, mapping input line and sample locations to output line locations, can be solved by swapping output line locations for output sample locations in the matrix [b]. The reverse mapping equations, mapping output locations to input line and sample, can be found by using output line and sample locations in the [A] matrix and the corresponding input sample and then line locations in the [b] matrix.

c).1. Calculate Forward Mappings
Using the Calculate Map Coefficients algorithm described above, generate the mapping polynomial coefficients needed to convert from a line/sample in input space (satellite) to one in output space (projection). Coefficients for every cell in the grid are generated.

c).2. Calculate Inverse Mappings
Using the Calculate Map Coefficients algorithm described above, generate the mapping polynomial coefficients needed to convert from a line/sample in output space (projection) to one in input space (satellite). Coefficients for every cell in the grid are generated.

4.2.2.6.2.5 Stage 5 - Finalize the Grid

The final stage of grid processing generates the global (rough) mapping coefficients, used to initially identify the appropriate grid cell, and computes the parallax sensitivity coefficients, used to correct for even/odd detector offset effects, for each grid cell.

a) Calculate Rough Mapping Coefficients
Calculate the rough mapping coefficients for the grid. The rough polynomial is a set of polynomials used to map output line and sample locations to input line and sample locations. The rough polynomial is generated using a large number of points distributed over the entire scene, and by calculating a polynomial equation that maps an output location to an input location. The rough polynomial is only meant to get a “close” approximation to the input line and sample location for a corresponding output line and sample location. Once this approximation is made, the value can be refined to get a more accurate solution. A rough mapping polynomial is found for every SCA, for every band, and for every elevation plane that is stored in the grid file.

Bilinear mapping was found to be sufficient for the rough mapping. Therefore, the mapping function looks like the ones used for each individual grid cell. However, the setup of the matrices to solve for the mapping coefficients is different:

$$\underset{Nx4}{[A]}\underset{4x1}{[coeff]} = \underset{Nx1}{[b]}$$

Where the matrix [A] is defined by the output line and sample locations, matrix [b] is defined by either the input lines or input samples, and N is equal to the total number of points stored in the grid for one elevation plane, of one band, for a single SCA. Therefore, the rough polynomial is found by using all the point locations stored in the grid for a given band and elevation plane for a single SCA. There is one mapping for

output line and sample location to input sample location, and one mapping for output line and sample location to input line location.

Grid Cell Polynomial
Calculate a "rough" mapping of output to input lines/samples. These coefficients are used as a first-order approximation to an inverse line-of-sight model. This polynomial is used to initially locate the grid cell to be used in the resampling process, providing a starting point for the more accurate inverse model based on individual grid cell parameters.

b) Calculate Detector Offsets
Computes the detector offset values and stores linear mapping coefficients associated with detector offsets in the grid structure. Using the zero elevation plane, for each band and each SCA, loop on the input lines and samples, calculating the odd/even detector offsets. The detector offsets are set up to account for the geometric differences between the odd/even, secondary, and tertiary detectors and the “nominal” set of detectors. (See the LOS Model Creation Algorithm (4.2.1)). These differences are considered to be consistent between actual and nominal detectors when they occur under the same acquisition conditions, i.e., they are slowly varying. These actual to nominal detector differences are due to the imperfect trade-off between space (detector offset) and time (detector delay) that is made when we temporally shift (through the use of Level 1R image fill) the even/odd and deselected detectors to compensate for their spatial offsets on the focal plane. The degree to which this time/space trade is imperfect varies with height, and so the corrections derived here and stored in the grid structure are functions of detector offset and height.

The subpixel detector-specific offsets are stored in the CPF. These "exact" detector-specific offsets are accounted for in the resampling process. Note that the potential for deselected detectors has made it necessary to also store per-detector full-pixel offsets in the CPF (and LOS model). As a result, this detector offset sensitivity logic has been changed to compute the offset sensitivity per pixel of detector offset rather than a fixed value derived from the static even/odd detector offset. The Image Resampling Algorithm discusses the routine ols2ils listed below, used for mapping an output line and sample to an input line and sample using the geometric grid.

Loop on the number of bands stored in the grid
Loop on the number of SCAs stored in the grid
Loop on the lines and samples stored in the grid

Get the maximum detector offset value for this band from the CPF.

Calculate the output line/sample location for the current input line and sample and the zero elevation plane, calculated using the forward model (see below) with the detector location set to

MAXIMUM. This detector type is the same as ACTUAL but uses the maximum detector offset rather than the detector-specific value.

Map the calculated output line/sample back to input space using the geometric grid and ols2ils.

Delta line/sample per pixel of offset are calculated as follows:

$\Delta line_0$ = (nominal line - mapped line) / max offset
$\Delta sample_0$ = (nominal sample - mapped sample) / max offset

where:
nominal line = current grid cell line location
mapped line = input line location from ols2ils mapped "maximum" output line
nominal sample = current grid cell sample location
mapped sample = input sample location from ols2ils mapped "maximum" output sample
max offset = detector offset used in the MAXIMUM forward model calculations

These delta lines and samples represent the input space correction necessary to compensate for the difference between nominal and actual detectors per pixel of detector offset, for the zero elevation plane.

Repeat these calculations for the maximum elevation plane to compute $\Delta line_H$ and $\Delta sample_H$, where H is the elevation corresponding to the maximum z-plane.

Compute the line and sample even/odd offset sensitivity coefficients:

$c_0 = \Delta line_0$
$c_1 = (\Delta line_H - \Delta line_0) / H$
$d_0 = \Delta sample_0$
$d_1 = (\Delta sample_H - \Delta sample_0) / H$

Note that c_0 and d_0 are in units of pixels per pixel and c_1 and d_1 are in units of pixels per meter per pixel.

These c_i and d_i coefficients are stored in the projection grid to be used during the resampling process.

Output Line/Sample to Input Line/Sample
Map output space line/sample locations back into its corresponding input space line/sample locations. This is done using the "rough" polynomial from the grid to determine an initial guess at an input space line and sample. From this initial guess, a grid cell row and column is calculated and the inverse coefficients for that cell are retrieved from the grid. These coefficients are used to determine an exact input space line and sample (in extended space).

Find Grid Cell
This utility function finds the correct cell that contains the output line/sample. It finds the correct grid cell containing the output pixel by first determining the set of grid cells to be checked. It then calls a routine to perform a "point in polygon" test on each of these grid cells to determine if the pixel does indeed fall within that grid cell.

Forward Model
Having described the grid generation procedure, we now turn to the forward model, referred to extensively above, in more detail.

For a given line, sample and band, propagate the forward model to determine a latitude and longitude for the specified point. This involves finding the time of the observation, constructing the instrument line-of-sight, calculating the spacecraft attitude and ephemeris for the observation time, and intersecting the projected line-of-sight with the Earth’s surface. The entire forward model procedure, referred to as LOS projection, is described step-by-step below.

a) Project LOS
Find the position where the line-of-sight vector intersects the Earth's surface. It invokes the following sub-algorithms:

a).1. Find Time
Find the time into the scene, given the line, sample, and band. The input sample number is 0-relative and relative to the SCA. The accounting for the odd/even, secondary, and tertiary detector offsets is based on the value of the dettype variable, which may be NOMINAL, ACTUAL, MAXIMUM or EXACT. Note that the EXACT selection is treated the same as ACTUAL. This is because fractional-pixel detector offsets can occur, but the compensating time shifts implemented by inserting fill pixels can only be introduced in whole-line increments. Therefore, the subpixel difference between the ACTUAL and EXACT detector types affects only the LOS angle, not the time. The MAXIMUM detector type represents a theoretical offset that is used to calculate the odd/even offset, or parallax, coefficients within the grid.

Due to the staggered odd/even and multiple pixel select detectors, a nominal and an actual time can be found in a scene. If the current position within the image is given as a line and sample location, the two different “types” of times for multispectral pixels are calculated as follows:

if detector type is set to MAXIMUM

```
            detector_shift_x = maximum_detector_shift
            l0r_fill_pixels = round(detector_shift_x) + nominal_fill
 else
            detector_shift_x = shift stored in the geometric model
             l0r_fill_pixels = Fill from L0Rp (also stored in the geometric model)

time_index = MS_line - l0r_fill_pixels
if ( time_index < 0 ) time_index = 0
if (time_index > (num_time_stamps - 1)) time_index = num_time_stamps - 1

MS_actual_time = line_time_stamp[time_index] - MS_settle_time  -
MS_integration_time/2
                    + (MS_line - l0r_fill_pixels - time_index) * MS_sample_time

MS_nominal_time = MS_actual_time + (l0r_fill_pixels – nominal_fill) *
MS_sample_time
```

where:

- MS_line is the zero-referenced multispectral line number (N).
- l0r_fill_pixels is the total amount of even/odd detector alignment fill to be inserted at the beginning of the pixel column associated with the current detector. This table is stored in the LOS model.
- num_time_stamps is the total number of time codes (data frames) in the image. It is tested to ensure that time_index, the line_time_stamp index, does not go out of bounds.
- detector_shift_x (unless type is MAXIMUM) is the amount of even/odd detector offset for the current detector from the LOS model detector delay table. It is rounded to the nearest integer pixel because time offsets can only occur in whole line increments. This detector shift is stored within the geometric model.
- MS_settle_time is a small sample and hold time delay constant.
- nominal_fill is the nominal fill associated with current band and SCA.
- maximum_detector_shift is the theoretical offset used in calculating the geometric effects associated with the odd/even offset of the detectors.

The MS_settle_time correction is expected to be a small (tens of microseconds) constant offset that should be captured in the CPF. The detector_shift_x offset parameter from the LOS model detector delay table is rounded to include the effects of even/odd detector stagger and detector deselect, but not the detector-specific subpixel offsets.

For the panchromatic band, the corresponding equations for a pan detector in the two pan lines (2N and 2N+1) associated with MS line N are computed as follows:

```
if detector type is set to MAXIMUM
            detector_shift_x = maximum_detector_shift
```

l0r_fill_pixels = round(detector_shift_x) + nominal_fill
else
detector_shift_x = shift stored in geometric model
l0r_fill_pixels = Fill from L0Rp (also stored in geometric model)

time_index = floor((pan_line - l0r_fill_pixels)/2)
if (time_index < 0) time_index = 0
if (time_index > (num_time_stamps - 1)) time_index = num_time_stamps - 1

Pan_actual_time = line_time_stamp[time_index] - Pan_settle_time - Pan_integration_time/2
+ (pan_line - l0r_fill_pixels - 2*time_index)*Pan_sample_time

Pan_nominal_time = Pan_actual_time + (l0r_fill_pixels – nominal_fill) * Pan_sample_time

where:

- pan_line is the zero-referenced panchromatic line number (2N or 2N+1).
- l0r_fill_pixels is the total amount of even/odd detector alignment fill to be inserted at the beginning of the pixel column associated with the current detector. These values are stored in the LOS model. Note that these values will always be even for the panchromatic band.
- num_time_stamps is the total number of time codes (data frames) in the image. It is tested to ensure that time_index, the line_time_stamp index, does not go out of bounds.
- detector_shift_x (unless type is MAXIMUM) is the amount of even/odd detector offset for the current detector from the LOS model detector delay table. It is rounded to the nearest integer pixel because time offsets can only occur in whole line increments. This detector shift is stored within the geometric model.
- Pan_settle_time is a small sample and hold time delay constant.
- nominal_fill is the nominal fill associated with the current band and SCA.
- maximum_detector_shift is the theoretical offset used in calculating the geometric effects associated with the odd/even offset of the detectors.

For the panchromatic band, the l0r_fill_pixels and detector_shift_x parameters are in units of panchromatic pixels.

a).2. Find LOS

Find the line-of-sight vector in sensor coordinates, using the Legendre polynomial LOS model stored in the LOS model, as follows:

Find the normalized detector for the Legendre polynomial:

$$\text{normalized detector} = \frac{2 * (\text{current detector})}{(\text{number of detectors} - 1)} - 1$$

where:

current detector = sample location (in the range 0 to number of detectors-1)

number of detectors = number of detectors (samples) for current band and SCA (from LOS model)

Find across-track (y) and along-track (x) angles:

$$x = coef\_x_0 + coef\_x_1 * (\text{normalized detector}) + coef\_x_2 * (1.5 * (\text{normalized detector})^2 - 0.5)$$

$$y = coef\_y_0 + coef\_y_1 * (\text{normalized detector}) + coef\_y_2 * (1.5 * (\text{normalized detector})^2 - 0.5)$$

where:

coef_x = Legendre coefficients for the along-track direction

coef_y = Legendre coefficients for the across-track direction

(Note: coef_x and coef_y are read from the CPF and stored in the LOS model)

If LOS requested is ACTUAL, add the whole pixel detector shift (detector, band, and SCA dependent for OLI) from the LOS model. This detector shift is only in the along-track direction. Note that the LOS model contains the combined whole pixel and subpixel detector offset, so it must be rounded to the integer part for the ACTUAL detector type and left unrounded for the EXACT detector type.

x = x + round(detector_shift_x) * IFOV

If LOS requested is EXACT, then add individual detector offsets (detector number, band, and SCA dependent). This detector shift is in both the along- and across-track directions. These values are stored within the LOS model.

x = x + (detector_shift_x) * IFOV
y = y + (detector_shift_y) * IFOV

Note that the detector_shift_y parameter, from the LOS model detector delay table, is always subpixel. See the LOS Model Creation Algorithm (4.2.1) for a further explanation of NOMINAL/ACTUAL/EXACT line of sight.

If the LOS request is MAXIMUM, then add the maximum, or theoretical, detector offset.

x = x + (maximum_detector_shift_x) * IFOV

Calculate LOS vector.

$$[\text{los}] = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Normalize LOS.

$$\vec{\text{los}} = \frac{\vec{\text{los}}}{\left\|\vec{\text{los}}\right\|}$$

a).3. Find Attitude

Find the precise roll, pitch, and yaw at the specified time. This routine uses the "corrected" version of the attitude data stored in the OLI LOS model. This attitude data sequence includes the effects of ground control point precision correction (if any).

Find the current time relative to attitude data start time stored in the LOS model.

dtime = time + image epoch time – attitude epoch time

Note:

time = nominal time of input sample relative to the start of the image epoch time = image start time from LOS model, only need seconds of day field since all epochs are adjusted to the same day. attitude epoch time = attitude data start time from LOS model, only need seconds of day field since all epochs are adjusted to the same day.

Find index into attitude data (stored in model) corresponding to dtime:

$$\text{index} = \text{floor}\left(\frac{\text{dtime}}{\text{attitude sampling rate}}\right)$$

where:

attitude sampling rate = sample period from LOS model

This attitude index determination could also be implemented as a search through the attitude data time stamps, which are stored in the LOS model. The selected index would be the index of the last time that does not exceed dtime.

Attitude is found by linearly interpolating between the attitude values located at index and index+1, using the corrected attitude sequence from the LOS model:

$$w = \frac{\text{fmod}\left(\text{dtime, attitude sampling rate}\right)}{\text{attitude sampling rate}}$$

$$\text{roll} = \text{model roll}_{index} + \left(\text{model roll}_{index+1} - \text{model roll}_{index}\right) * w$$

$$\text{pitch} = \text{model pitch}_{index} + \left(\text{model pitch}_{index+1} - \text{model pitch}_{index}\right) * w$$

$$\text{yaw} = \text{model yaw}_{index} + \left(\text{model yaw}_{index+1} - \text{model yaw}_{index}\right) * w$$

a).3.i. Find Jitter

Find the high-frequency roll, pitch, and yaw corrections at the specified input image line/sample coordinate. This routine uses the jitter table stored in the OLI LOS model. This table is time aligned with the OLI panchromatic band line sample times, so the jitter table look-up proceeds directly from the input line/sample coordinates:

Find the current detector number from the input sample location:

detector = round(sample)

Verify that the detector is in the valid range for this band (return error if not).

Look up the number of L0R fill pixels for this detector (from the fill table).

Calculate the jitter table index:

If (band = pan)

Index = round(line) – l0r_fill_pixels

Else

Index = 2*(round(line) – l0r_fill_pixels)

Verify that jitter table index is within the valid range for the table (return zeros if not).

Extract the roll-pitch-yaw jitter values for the current index from the jitter table and return these values.

Note that the jitter values are a direct look-up without interpolation. This does not compromise accuracy because this function is only used for cases of EXACT detector projection (e.g., the OLI data simulator) for which the input line/sample coordinates are integers. The jitter values extracted by Find Jitter are added to the low-frequency roll-pitch-yaw values interpolated by Find Attitude by the calling procedure Get LOS when the EXACT option is in force.

a).4. Move Satellite Sub-Algorithm

Compute the satellite position and velocity at a delta time from the ephemeris reference time using Lagrange interpolation. This is a utility sub-algorithm that accesses the "corrected" version of the model ephemeris data to provide the OLI position and velocity at any specified time. Since the model ephemeris arrays are inputs to this sub-algorithm, it will work with either the ECI or ECEF ephemeris data.

Calculate the time of the current line/sample relative to the start time of the ephemeris start time.

reference time = time + image epoch time – ephemeris epoch time

where:

time = nominal time of input sample relative to the start of the imagery

image epoch time = image start time from LOS model, only need seconds of day since all epochs are on same day.

ephemeris epoch time = ephemeris start time from LOS model, only need seconds of day since all epochs are on same day.

Find index into ephemeris data stored in model.

$$\text{index} = \text{floor}\left(\frac{\text{reference time}}{\text{ephemeris time steps}} - \frac{\text{number of Lagrange points}}{2} \right)$$

where:

ephemeris time steps = time between ephemeris samples
number of Lagrange points = number of points to use in Lagrange interpolation

Use Lagrange interpolation to calculate satellite position and velocity in ECEF (or ECI, depending on which sequence is provided) coordinates at time of current line/sample.

X = Lagrange(model satellite ECEF/ECI x[index])
Y = Lagrange(model satellite ECEF/ECI y[index])
Z = Lagrange(model satellite ECEF/ECI z[index])
XV = Lagrange(model satellite ECEF/ECI vx[index])
YV = Lagrange(model satellite ECEF/ECI vx[index])
ZV = Lagrange(model satellite ECEF/ECI vx[index])

where:

X = satellite x coordinate
Y = satellite y coordinate
Z = satellite z coordinate
XV = satellite x velocity
YV = satellite y velocity
ZV = satellite z velocity

a).5. Convert Sensor LOS to Geocentric

Find the line-of-sight vector from the spacecraft to a point on the ground by transforming the line-of-sight vector in sensor coordinates to perturbed spacecraft coordinates.

Use the OLI alignment matrix in the LOS model to convert the LOS vector from sensor to body. Then, apply roll, pitch, and yaw to the LOS to convert body to orbital. Finally, use the ephemeris to construct the orbital to ECEF rotation matrix and use it to transform LOS to ECEF.

First, using the 3x3 ACS to instrument alignment transformation matrix stored in the LOS model, calculate the instrument to ACS transformation matrix.

$$[\text{ Instrument to ACS }] = [\text{ ACS to Instrument }]^{-1}$$

Transform LOS from Instrument to ACS/body coordinates.

$$[\text{ navigation los }] = [\text{ Instrument to ACS }]\ [\text{ los }]$$

Calculate the attitude perturbation matrix, using interpolated attitude values. Note that these values include the effects of precision LOS correction (if any), as these will be built into the "corrected" attitude stream in the LOS model. For the Earth-view acquisitions, the roll-pitch-yaw values will be with respect to the orbital coordinate system, but for celestial acquisitions, they will be with respect to ECI.

Calculate the perturbation matrix, [perturbation], due to roll, pitch, and yaw:

$$[\text{ perturbation }] = [\ Y_{yaw}\ ]\ [\ P_{pitch}\ ]\ [\ R_{roll}\ ] =$$

$$\begin{bmatrix} \cos(p)\cos(y) & \sin(r)\sin(p)\cos(y)+\cos(r)\sin(y) & \sin(r)\sin(y)-\cos(r)\sin(p)\cos(y) \\ -\cos(p)\sin(y) & \cos(r)\cos(y)-\sin(r)\sin(p)\sin(y) & \cos(r)\sin(p)\sin(y)+\sin(r)\cos(y) \\ \sin(p) & -\sin(r)\cos(p) & \cos(r)\cos(p) \end{bmatrix}$$

Calculate the new LOS in orbital coordinates (Earth-view) or ECI (celestial) due to attitude perturbation:

[perturbation los] = [perturbation] [navigation los]

For Earth-view acquisitions, calculate the transformation from Orbital Coordinates to ECEF. The position and velocity vectors used in calculating the transformation are those calculated above. These vectors are in ECEF, allowing the LOS to be transformed from the instrument coordinate system to the ECEF coordinate system.

Transform perturbed LOS from Orbital to ECEF.

[ECEF los] = [ORB2ECEF] [perturbation los]

For celestial acquisitions, the ECI LOS ([perturbation los]) is returned.

a).6. Find Target Position

Finds the position where the line-of-sight vector intersects the Earth's surface.

Intersect the LOS in ECEF with the Earth model calculating the target ECEF vector. The ECEF vector is then used to compute the geodetic latitude and the longitude of the intersection point.

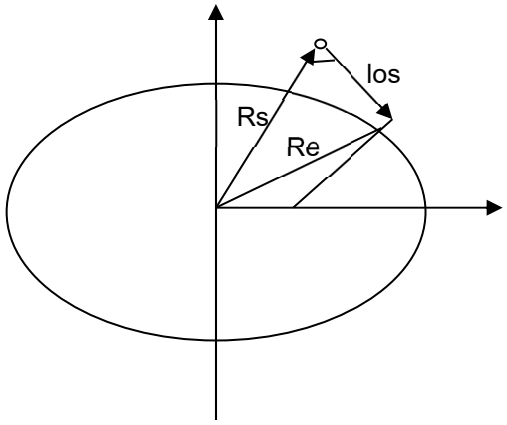

Figure 4-29. Intersecting LOS with Earth model

Where:

Rs = satellite position vector
Re = geocentric Earth vector
los = line-of-sight vector

Intersect LOS with ellipsoid

1. Rescale vectors with ellipsoid parameters.

$$rs' = \left[\frac{rsx}{a} \quad \frac{rsy}{a} \quad \frac{rsz}{b} \right]$$
$$re' = \left[\frac{rex}{a} \quad \frac{rey}{a} \quad \frac{rez}{b} \right]$$
$$los' = \left[\frac{losx}{a} \quad \frac{losy}{a} \quad \frac{losz}{b} \right]$$

where:

a = semi-major axis of Earth ellipsoid
b = semi-minor axis of Earth ellipsoid
rs' = rescaled satellite position vector
re' = rescaled geocentric Earth vector
los' = rescaled LOS vector

2. From the Law of Cosines

$$|re'|^2 = |d * los'|^2 + |rs'|^2 - 2|d * los'||rs'| \cos(\delta)$$

where:

d = los' vector length
δ = angle between rs' and los'

$$\cos(\delta) = \frac{los' \bullet rs'}{|los'||rs'|}$$

By definition | re' | = 1

Rearranging the equation determined from the Law of Cosines in terms of the constant d.

$$d^2|los'|^2 + 2d(los' \bullet rs') + |rs'|^2 - 1$$

Solving for d using the quadratic equation.

$$d = \frac{-|los' \bullet rs'| - \sqrt{|los' \bullet rs'|^2 - |los'|^2\left(|rs'|^2 - 1\right)}}{|rs'|^2}$$

3. Compute the new target vector.

$$re' = rs' + d * los'$$

4. Rescale the target vector.

$$re = [a * rex' \quad a * rey' \quad b * rez']$$

5. Compute the Geodetic coordinates (see Geocentric to Geodetic below).

$$(rex, rey, rez) \Rightarrow (\phi_0, \lambda_0, h_0)$$

If target height (H), or elevation corresponding to current z plane, is not zero:

Initialize:
Target vector: rt=re
Target height: h_0=0

Iterate until $\Delta h = (h_i - H)$ is less than TOL

a) Calculate the delta height.

$\Delta h = h_i - H$

b) Compute the length of LOS.

$$d = \sqrt{(rtx - rsx)^2 + (rty - rsy)^2 + (rtz - rsz)^2}$$

where:
d = length of the LOS vector
rt = target vector
rs = spacecraft position vector

c) Compute the LOS /height sensitivity.

$$q = n \bullet los$$

Where **n** is a vector normal to the ellipsoid surface.

$$n = [cos(\phi_i)\, cos(\lambda_i) \quad cos(\phi_i)\, sin(\lambda_i) \quad sin(\lambda_i)]^T$$

and:
q = LOS height sensitivity coefficient
los = LOS unit vector
ϕ_i = current estimate of ground point latitude
λ_i = current estimate of ground point longitude

d) Adjust LOS.

$$d = d + q * \Delta h$$

e) Re-compute the target vector.

$$rt = rs + d * los$$

f) Calculate the new geodetic coordinates and corresponding height above ellipsoid.

$$(rtx, rty, rtz) \Rightarrow (\phi_{i+1}, \lambda_{i+1}, h_{i+1})$$

Calculate the geodetic latitude and longitude from the final ECEF vector.

a).7. Geocentric to Geodetic

The relationship between ECEF and geodetic coordinates can be expressed simply in its direct form:

$e^2 = 1 - b^2 / a^2$
$N = a / (1 - e^2 \sin^2(\varphi))^{1/2}$
$X = (N + h) \cos(\varphi) \cos(\lambda)$
$Y = (N + h) \cos(\varphi) \sin(\lambda)$
$Z = (N (1-e^2) + h) \sin(\varphi)$

where:

X, Y, Z	-	ECEF coordinates
φ, λ, h	-	Geodetic coordinates (lat ϕ, long λ, height h)
N	-	Ellipsoid radius of curvature in the prime vertical
e^2	-	Ellipsoid eccentricity squared
a, b	-	Ellipsoid semi-major and semi-minor axes

The closed-form solution for the general inverse problem (which is the problem here) involves the solution of a quartic equation, and is not typically used in practice. Instead, an iterative solution is used for latitude and height for points that do not lie on the ellipsoid surface, i.e., for h ≠ 0.

To convert ECEF Cartesian coordinates to spherical coordinates:

Define:

$$\text{radius} = \sqrt{X^2 + Y^2 + Z^2}$$

$$\varphi' = \sin^{-1}\left(\frac{Z}{\text{radius}}\right)$$

$$\lambda = \tan^{-1}\left(\frac{Y}{X}\right)$$

Initialize:

$$\theta = \varphi'$$

$$h_0 = 0$$

Iterate until abs(h_i-h_{i+1}) < TOL

$$\text{re} = \frac{a * \sqrt{1-e}}{\sqrt{1-e * \cos^2(\theta)}}$$

$$\varphi = \tan^{-1}\left(\frac{\tan(\theta)}{1-e}\right)$$

$$\Delta\varphi = \varphi - \theta$$

$$\text{rs} = \text{radius}^2 - \text{re}^2 * \sin^2(\Delta\varphi)$$

$$\text{h}_{i+1} = \sqrt{\text{rs}} - \text{re} * \cos(\Delta\varphi)$$

$$\theta = \varphi' - \sin^{-1}\left(\frac{\text{h}_{i+1}}{\text{radius} * \sin(\Delta\varphi)}\right)$$

Projection Transformation

Convert coordinates from one map projection to another. The transformation from geodetic coordinates to the output map projection depends on the type of projection selected. The mathematics for the forward and inverse transformations for the UTM, Lambert Conformal Conic, Transverse Mercator, Oblique Mercator, Polyconic, and the Space Oblique Mercator (SOM) map projections are handled by USGS's General Cartographic Transformation Package (GCTP).

Grid Structure Summary

Table 4-8 and Table 4-9 show the detailed contents of the geometric grid structure.

Geometric Grid Structure Contents
Satellite Number (8)
WRS Path
WRS Row (may be fractional)
Acquisition Type (Earth, Lunar, Stellar)
Scene Framing Information:
Frame Type: PROJBOX, MAXBOX, PATH_MAXBOX, LUNAR, or STELLAR
Projection Units (text): METERS, RADIANS, ARCSECONDS
Projection Code: GCTP integer code for UTM, SOM, etc...
Datum: WGS84
Spheroid: GCTP integer code = 12 (WGS84/GRS80)
UTM Zone: UTM zone number (or 0 if not UTM)
Map Projection Parameters: 15-element double array containing parameters
Corners: 4 by 2 array of projection coordinates for UL, LL, UR, and LR corners
Path-oriented Framing Information:
Center Point: latitude and longitude of WRS scene center
Projection Center: Map x/y of WRS scene center
Rotation Angle: Rotation (from true north) of the path frame (degrees)
Orientation Angle: Rotation (from grid north) of the path frame (degrees)
Active Image Areas: latitude and longitude (in degrees) of the four corners of the active image area (excluding leading and trailing SCA imagery) for each band
Grid Structure Information:
Number of SCAs
Number of Bands
Band List: array of band IDs included in grid
Array of band grid structures, one for each SCA in each band (see Table 4-9)

Table 4-8. Geometric Grid Structure Contents

Grid Structure Contents for Each SCA in Each Band
Band number
Grid cell size: number of image lines and samples in each grid cell
Grid cell scale: 1/lines per cell and 1/samples per cell
Pixel size: in projection units (usually meters)
Number of lines in the output image
Number of samples in the output image
Number of lines in the grid (NL)
Number of samples in the grid (NS)
Number of z-planes (NZ)
Index of zero-elevation z-plane
Z-plane spacing: elevation increment between z-planes

1D array of input line numbers corresponding to each grid row
1D array of input sample numbers corresponding to each grid column
3D array of output lines for each grid point (row-major order) (NS*NL*NZ)
3D array of output samples for each grid point (row-major order) (NS*NL*NZ)
Array of line c_0, c_1 even/odd offset coefficients (row-major order) (2*NS*NL)
Array of sample d_0, d_1 even/odd offset coefficients (row-major order) (2* NS*NL)
3D array of forward mapping (ils2ols) coefficient sets (NS*NL*NZ)
3D array of inverse mapping (ols2ils) coefficient sets (NS*NL*NZ)
3D array of line jitter sensitivity coefficient vectors (note 2) (3*NS*NL*NZ)
3D array of sample jitter sensitivity coefficient vectors (note 2) (3*NS*NL*NZ)
Degree of rough polynomial
Array of rough line polynomial coefficients ((degree+1)^2 * NZ values)
Array of rough sample polynomial coefficients ((degree+1)^2 * NZ values)

Table 4-9. Per Band Geometric Grid Structure Contents

Geometric Grid Size

Fully capturing the potential variability of the 50 Hz attitude data that will be available within the L8/9 ancillary data stream would require a grid spacing of 5 lines. This may be impractical. Fortunately, the OLI error budgets assumed that attitude variations at frequencies up to only 10 Hz would be corrected in the LOS model. Such variations can be captured by sampling at 20 Hz or higher. This corresponds to a grid spacing of 11-12 lines. The grid has been successfully tested down to a line sampling of 10, but this does make for a large grid structure. The inclusion of a high-frequency jitter table in the OLI model and jitter sensitivity coefficients in the grid structure allow the grid to be less dense in the time (line) dimension. The baseline assumption is that attitude frequencies above 3 Hz will be relegated to the jitter table, allowing the grid density to be reduced to 30 lines, thus saving grid space even with the addition of the new jitter sensitivity fields.

4.2.2.7 Notes

The following are additional background assumptions and notes:

1. The NOVAS planetary ephemeris file provides the lunar ephemerides used to define the reference output space for lunar image processing. This file is in the original JPL format and is provided to the NOVAS routines as an input.
2. The TIRS implementation of the LOS projection grid will include another new feature that has also been applied to OLI as of version 3.4 of this algorithm – a set of sensitivity coefficients that maps roll-pitch-yaw deviations to the corresponding line and sample differences, for each grid cell. The resampler will use these sensitivity coefficients to convert high-frequency (per image line) attitude variations to line and sample adjustments. Modeling the high-frequency deviations separately and correcting them in the resampler allows for a sparser and more manageable-sized grid.

4.2.3 OLI Line-of-Sight Model Correction Algorithm

4.2.3.1 Background/Introduction

The LOS model correction algorithm uses the results of GCP measurements in an image that was systematically and terrain corrected using the original LOS model to

derive estimates for corrections to the model. Corrections to the spacecraft attitude and ephemeris are computed in a least squares procedure, which minimizes the differences between the measured locations of the control points and their true ground locations. This procedure includes the detection and removal of outlier GCPs, using a modified t-distribution test. These corrections are added to the LOS model structure to create a "precision" model for subsequent use by other geometric algorithms. Creating the precision model involves repeating some of the processing originally performed by the LOS model creation algorithm to incorporate the model corrections. Once the precision model corrections are computed, the algorithm performs simple threshold tests (e.g., on the pre-fit and post-fit Root-Mean Squared (RMS) GCP residuals and the percentage of GCPs declared outliers) to determine if the solution was successful. If the solution is not successful, the LOS model is not updated with the corrections.

4.2.3.2 Dependencies

The LOS Model Correction algorithm assumes that GCPs exist for the ground site and that the Model Creation, LOS Projection and Gridding, and Image Resampling algorithms have been executed to create a systematic terrain-corrected image for GCP mensuration. Note that the band selection and resolution of this mensuration image will depend on the flow being executed/control source being used. For standard L1T product generation, the GLS control (SWIR1 band, 30m resolution) will be used, whereas for characterization and calibration flows, the DOQ control (panchromatic 15m) will be used. It further assumes that the GCP Correlation algorithm/utility has been executed to measure the GCP locations in the mensuration image. The mensuration image may be either SCA-separated or SCA-combined, though SCA-combined images will be the preferred mode of operation.

4.2.3.3 Inputs

The LOS Model Correction algorithm uses the inputs listed in the following table. Note that some of these "inputs" are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data; including dataset IDs to provide unique identifiers for data trending).

Algorithm Inputs
ODL File (implementation)
Measured GCP file name
OLI LOS model file name
OLI grid file name
DEM file name
CPF file name
L1G image file name
Precision solution parameters:
Apriori weights for attitude correction parameters (in microradians and microradians/second)
Apriori weights for ephemeris correction parameters (in meters and meters/second)
Correction model parameterization options (att_orb, eph_yaw, both, weight - default is "both")
Bias correction or rate of change correction option (time flag)
Apriori weights for GCP measurements (in at-sensor microradians)

Algorithm Inputs
Iteration limit
Outlier threshold
Processing Options (implementation):
Residual Trending On/Off Switch
Solution/Alignment Trending On/Off Switch
L0Rp ID (for trending)
Work Order ID (for trending)
Measured GCP File Contents (see GCP Correlation Algorithm Section 4.1.6 for details)
GCP image positions
GCP ground coordinates
OLI Grid File Contents (see LOS Projection Algorithm Section 4.2.2 for details)
Arrays of Input/Output Mappings
Output Image Frame (e.g., corners, map projection)
OLI LOS Model Contents (see LOS Model Creation Algorithm Section 4.2.1 for details)
WRS Path/Row
Number of input image (L1R) lines
Smoothed image time codes
Integration Time (pan and multispectral bands)
Smoothed ephemeris at 1 second intervals
Earth orientation parameters (UT1UTC, pole wander)
OLI to ACS reference alignment matrix/quaternion
Spacecraft CM to OLI offset in ACS reference frame
Focal plane model parameters (number of SCAs, number detectors/band, Legendre coefficients)
Detector offset table (including detector deselect offsets)
CPF File Contents
Pre-fit RMS threshold
Post-fit RMS threshold
Percent outlier threshold
Minimum number of valid GCPs threshold
L1G Image File Contents
L1G Metadata
DEM File Contents (see notes 9 and 10)
DEM Metadata
Elevation Data

4.2.3.4 Outputs

Precision LOS Model (only items that are updated from the input LOS model are listed below)
Updated corrected ephemeris at 1 second intervals
Updated corrected attitude data sequence
Precision correction reference date/time
Precision attitude and ephemeris corrections
LOS Model Correction Solution File (see Table 4-10 below for additional details)
LOS model correction reference date/time
Final iteration precision correction values
Final iteration precision correction covariance
LOS Model Correction Residuals File (see Table 4-11 below for additional details)
GCP residuals for each point for each iteration

Correction Solution and Alignment Trending Data (see Table 4-12 below for additional details)
Precision correction reference date/time
Precision attitude/ephemeris correction values (see note 1)
Reduced precision correction covariance (see note 2)
Solution quality metrics (see note 4)
Control type used (GLS or DOQ)
Off-nadir angle (in degrees)
L0Rp ID
Work Order ID
WRS Path/Row
Correction Residuals Trending Data (see note 3 and Table 4-13 below for details)
WRS Path/Row
GCP ID
GCP Type (GLS or DOQ)
Date/Time of imaging
Spacecraft position/velocity at image time
GCP ground coordinates (lat,lon,height)
Apparent GCP position (lat, lon, height) in mensuration image
LOS Model Correction Success/Failure Status Return

4.2.3.5 Options

Solution/Alignment Trending On/Off Switch
Residual Trending On/Off Switch

4.2.3.6 Procedure

The LOS correction procedure uses the GCP measurements collected by the GCP Correlation algorithm to estimate updates to the spacecraft attitude and ephemeris data, which minimize the discrepancies between the actual (known) GCP locations and the apparent locations measured in the terrain-corrected L1G image. The solution method adopted for OLI is essentially the same as that used for Landsat 7 and for the ALI, wherein "truth" and "observed" LOS vectors are constructed in the orbital coordinate system, and a weighted least-squares solution is used to minimize the misalignments between the truth and observed vectors. The solution supports the estimation of offset and rate corrections for all three ephemeris position axes and for all three attitude angles (roll-pitch-yaw), though options are provided to reduce the number of parameters (e.g., solve for offsets only) to accommodate situations where the GCPs are few in number and/or poorly distributed.

There are several differences in the implementation of the OLI LOS correction model as compared to the previous missions. The first is a change in the coordinate system in which the corrections are applied. For Landsat 7 and ALI data, the precision corrections were applied in the orbital coordinate system. For OLI, they are applied in the spacecraft body/attitude control system coordinate system (this was also noted in the Ancillary Data Preprocessing Algorithm Section 4.1.4. For nadir-viewing scenes, there is little difference, but the case of off-nadir viewing leads to a few adjustments to the heritage algorithm in what follows.

The second significant difference is in the way that the corrections are reflected in the precision LOS model created as an output by this procedure. In the heritage implementation, the ephemeris corrections were used to update the model ephemeris data sequence, but the attitude corrections were stored as a separate correction model that was applied explicitly in the forward model. For OLI, corrected versions of both the ephemeris and attitude data sequences are computed using the LOS correction solution results. These corrected data are stored in the LOS model, along with the original ephemeris and attitude values. The parameters of the correction model are also included in the model, though they are there primarily for documentation purposes and are no longer used in the forward model computation.

The third difference is the inclusion of a portion of the sensor alignment calibration logic into the LOS correction algorithm. This logic uses the OLI to ACS alignment matrix stored in the OLI LOS model to convert the computed attitude offset corrections to OLI alignment angles. This yields updated estimates of the OLI to ACS alignment angles that are output to the characterization database for subsequent trending in the sensor alignment calibration procedure.

A fourth difference is the use of L1G terrain but not precision-corrected images to measure the control points (see the GCP Correlation Algorithm). This gives the apparent (measured) GCP location a non-zero height coordinate. The true GCP elevation (from the known GCP ground location) could be used, but it is more correct to interpolate the apparent point height from the DEM used to create the terrain-corrected L1G mensuration image. The use of terrain-corrected mensuration images also allows these images to be SCA-combined since the SCA overlap areas will be geometrically consistent.

The mathematical underpinnings of the LOS correction algorithm are presented first, followed by an overview of the procedure for implementing the algorithm.

Mathematical Development

The following subsections present the mathematical background of the LOS correction algorithm. In what follows, the equations presented are numbered so that they can be more easily referenced in the subsequent mathematical formulation and in the algorithm procedure sections.

1. Formulating the Observations

The geometric measurement in the OLI sensor system can be regarded as the look vector, $\boldsymbol{l}_{sc}$, in the spacecraft body-fixed system. This vector is transformed into the Orbit Reference Frame (OB) system (see Figure 4-30), as described in the Ancillary Data Preprocessing Algorithm Section 4.1.4, through the spacecraft attitude parameters:

$$\boldsymbol{l}_{ob} = \mathbf{T}^{\mathrm{T}}(\theta_r, \theta_p, \theta_y)\, \boldsymbol{l}_{sc}. \qquad (1.1)$$

where θ_r, θ_p, and θ_y are roll, pitch, and yaw angles, **T** is the transformation matrix, and can be expressed as

$$\mathbf{T}(\theta_r, \theta_p, \theta_y) = \mathbf{R}_3(\theta_y)\mathbf{R}_2(\theta_p)\mathbf{R}_1(\theta_r)$$

$$= \begin{bmatrix} \cos\theta_p\cos\theta_y & \cos\theta_r\sin\theta_y + \sin\theta_r\sin\theta_p\cos\theta_y & \sin\theta_r\sin\theta_y - \cos\theta_r\sin\theta_p\cos\theta_y \\ -\cos\theta_p\sin\theta_y & \cos\theta_r\cos\theta_y - \sin\theta_r\sin\theta_p\sin\theta_y & \sin\theta_r\cos\theta_y + \cos\theta_r\sin\theta_p\sin\theta_y \\ \sin\theta_p & -\sin\theta_r\cos\theta_p & \cos\theta_r\cos\theta_p \end{bmatrix}$$

$$\cong \begin{bmatrix} 1 & \theta_y & -\theta_p \\ -\theta_y & 1 & \theta_r \\ \theta_p & -\theta_r & 1 \end{bmatrix} \qquad (1.2)$$

where $\mathbf{R}_1$, $\mathbf{R}_2$, and $\mathbf{R}_3$ are the coordinate system transformation matrix for rotation around the x, y, and z-axis, respectively.

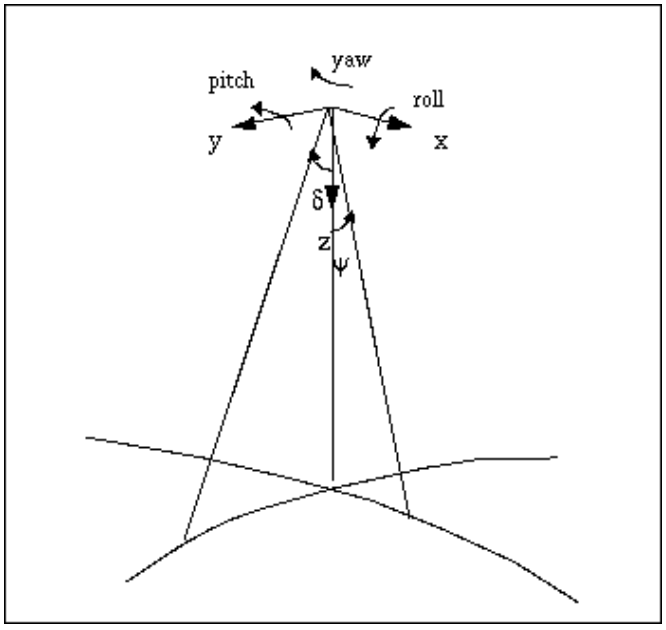

Figure 4-30. Definition of Orbit Reference System

The vector $\boldsymbol{l}_{ob}$ is further transformed into the ECF system

$$\boldsymbol{l}_{ef} = \mathbf{T}_f(\mathbf{r}_{ef}, \mathbf{v}_{ef})\, \boldsymbol{l}_{ob} \qquad (1.3)$$

where $\mathbf{T}_f$ is the forward transformation for vectors from the OB system to the ECF system, as a function of the satellite position $\mathbf{r}_{ef}$ and velocity $\mathbf{v}_{ef}$ vectors in the ECF system. Note that $\mathbf{v}_{ef}$ should be the "inertial" velocity expressed in the ECF system, as described in the Ancillary Data Preprocessing Algorithm Section 4.1.4. Vector $\boldsymbol{l}_{ef}$, together with the satellite position vector, $\mathbf{r}_{ef}$, is then used to intersect the ellipsoid Earth

surface to pin down a point position, $\mathbf{R}_{ef}$, as the target point on the Earth. This is the common forward image pixel geolocation calculation (forward model). Note that when using a terrain-corrected L1G mensuration image, the $\mathbf{R}_{ef}$ point represents the intersection of the LOS with the DEM used to create the L1G image, rather than the Earth ellipsoid surface. Thus, the target point will have a non-zero height coordinate.

Mathematically, $\mathbf{R}_{ef}$ is a function of $\boldsymbol{l}_{sc}$, θ_r, θ_p, θ_y, $\mathbf{r}_{ef}$, and $\mathbf{v}_{ef}$.

$$\mathbf{R}_{ef} = F(\boldsymbol{l}_{sc}, \theta_r, \theta_p, \theta_y, \mathbf{r}_{ef}, \mathbf{v}_{ef}) \tag{1.4}$$

Because of errors in the satellite orbit ephemeris and attitude data, this calculated $\mathbf{R}_{ef}$ is different from the true location of the image pixel. If we know the true location of a landmark pixel ($\mathbf{R}_{cp}$) from other sources (i.e., base map, survey), this point can be taken as a GCP to check the accuracy of the computed image pixel location. The precision correction process uses the GCP coordinates to estimate the correction to the satellite ephemeris and attitude data, so that with the corrected parameters in equation (1.4), the calculated image pixel location, $\mathbf{R}_{ef}$, will be close to its true location, $\mathbf{R}_{cp}$ (depending on the GCP positional accuracy).

To calculate the precision correction, the difference between $\mathbf{R}_{ef}$ and $\mathbf{R}_{cp}$ is taken as the observable, and the observation equation becomes:

$$d\mathbf{R} = \mathbf{R}_{cp} - F(\boldsymbol{l}_{sc}, \theta_r, \theta_p, \theta_y, \mathbf{r}_{ef}, \mathbf{v}_{ef}) \tag{1.5}$$

according to equation (1.4). However, the actual calculation of $\mathbf{R}_{ef}$ is usually not an explicit function of the orbit and attitude parameters, especially for the intersecting procedure. Therefore, it is inconvenient to linearize equation (1.5) with standard estimation techniques. Instead, the calculation of look vector $\boldsymbol{l}_{cp}$ corresponding to $\mathbf{R}_{cp}$, in the OB system, is much more explicit:

$$\mathbf{l}_{cp} = \mathbf{T}_i(\mathbf{r}_{ef}, \mathbf{v}_{ef}) \frac{(\mathbf{R}_{cp} - \mathbf{r}_{ef})}{|\mathbf{R}_{cp} - \mathbf{r}_{ef}|} \tag{1.6}$$

where ($\mathbf{R}_{cp}$ - $\mathbf{r}_{ef}$) is the LOS vector in the ECF system corresponding to $\mathbf{R}_{cp}$, and $\mathbf{T}_i(\mathbf{r}_{ef}$, $\mathbf{v}_{ef}$) is the inverse transformation for the look vector from the ECF system to the OB system. If all of the satellite attitude and ephemeris parameters are accurate, the $\boldsymbol{l}_{cp}$ from equation (1.6) and $\boldsymbol{l}_{ob}$ from equation (1.1) should be equal. Since the measurement $\boldsymbol{l}_{sc}$ is accurate compared to the attitude and ephemeris information, any systematic difference between $\boldsymbol{l}_{cp}$ and $\boldsymbol{l}_{ob}$ can be attributed to the attitude and orbit errors. Thus, we can use the difference between $\boldsymbol{l}_{cp}$ and $\boldsymbol{l}_{ob}$ as the observable.

$$d\mathbf{l} = \boldsymbol{l}_{cp} - \boldsymbol{l}_{ob} = \mathbf{T}_i(\mathbf{r}_{ef}, \mathbf{v}_{ef}) \frac{(\mathbf{R}_{cp} - \mathbf{r}_{ef})}{|\mathbf{R}_{cp} - \mathbf{r}_{ef}|} - \mathbf{T}(\theta_r, \theta_p, \theta_y)\, \boldsymbol{l}_{sc} \tag{1.7}$$

The task of precision modeling is then to calculate the correction to those satellite ephemeris and attitude parameters (i.e., $\mathbf{r}_{ef}$, $\mathbf{v}_{ef}$ and θ's) so that the residuals of d$\mathbf{l}$ after correction are minimized for all selected GCPs. The orbit correction is modeled as a linear function of time for each component in the OB system. Referred to as the short arc method, this purely geometric method shifts and rotates the short arc of orbit defined by the original ephemeris points to fit the GCP measurements.

2. Linearizing the Observations

These observation equations can be linearized with the following steps. In equation (1.7), the calculation of $\boldsymbol{l}_{ob}$ can also be carried out through the following:

$$\boldsymbol{l}_{ob} = T_i(\mathbf{r}_{ef}, \mathbf{v}_{ef})(\mathbf{R}_{ef} - \mathbf{r}_{ef}) / | \mathbf{R}_{ef} - \mathbf{r}_{ef} | \qquad (2.1)$$

if $\mathbf{R}_{ef}$ is more conveniently accessible. Since equation (2.1) is simply the inverse of equation (1.4) and equation (1.3), the $\boldsymbol{l}_{ob}$ calculated from equation (2.1) is the same as the one in equation (1.1), except for the possible inclusion of numerical errors. However, it should be mentioned that the true relationship between $\boldsymbol{l}_{ob}$ and the parameters is always equation (1.1). Equation (2.1) should not be confused with this, because $\mathbf{R}_{ef}$ in equation (2.1) is not an independent variable, but a function of equation (1.4). Therefore, in observation (1.7), information about the attitude parameters is contained in $\boldsymbol{l}_{ob}$ and the information about orbit parameters comes from $\boldsymbol{l}_{cp}$.

Since the measurement of $\boldsymbol{l}_{sc}$ is two-dimensional in nature, only two-dimensional information is contained in equation (1.7), although there are three components involved. If a look vector (either $\boldsymbol{l}_{cp}$ or $\boldsymbol{l}_{ob}$) has the three components in the OB system.

$$\mathbf{l} = \{xl, yl, zl\} \qquad (2.2)$$

The real information in these three components can be summarized in two variables like the original look angle measurements. We chose the following two variables:

$$\delta = \text{atan}\,(yl / zl) \qquad (2.3)$$

$$\psi = \text{atan}\,(xl / zl) \qquad (2.4)$$

So that the three components of equation (1.7) can be reduced to the two equations:

$$\alpha = \delta_{cp} - \delta_{ob} \qquad (2.5)$$

$$\beta = \psi_{cp} - \psi_{ob} \qquad (2.6)$$

Note that in equation (2.3) and (2.4), the components of *xl, yl,* and *zl* can be that of LOS vector instead of unit look vector, so that δ_{cp} and ψ_{cp} are explicit functions of orbit position. In that case, *zl* is approximately the height of the satellite.

If we define,

true value = approximate value + correction

and differentiate equations (2.3) and (2.4) with respect to the orbit position (for δ_{cp} and ψ_{cp}), differentiate equation (1.1) with respect to the satellite attitude (for δ_{ob} and ψ_{ob}) at their corresponding approximate values, then equations (2.5) and (2.6) can be linearized as the function of correction parameters.

$$\alpha \cong (\cos^2\delta_{cp} / h)\, dy - (\cos\delta_{cp} \sin\delta_{cp} / h)\, dz + d\theta_r \qquad (2.7)$$

$$\beta \cong (1.0 / h)\, dx - d\theta_p + \tan\delta_{cp}\, d\theta_y \qquad (2.8)$$

where dx, dy, and dz are the correction to satellite position vector $\mathbf{r}_{ob}$ in the OB system, and $d\theta$'s are the corrections to the satellite attitude angle θ's. Other quantities are functions evaluated at the approximate values of $\mathbf{r}_{ef}$, $\mathbf{v}_{ef}$, and θ's.

The linearization above is done by directly differentiating equation (2.3) and (2.4), with transformation $\mathbf{T}_i$ regarded unaffected by the error in $\mathbf{r}_{ef}$ and $\mathbf{v}_{ef}$. This, however, ignores the curvature of the satellite orbit and the Earth, resulting in about 10 percent of error in the coefficients of dx, dy, and dz. A more accurate way to evaluate these coefficients is to examine the sensitivity terms $d\psi_{cp}/dx$, $d\delta_{cp}/dy$, and $d\delta_{cp}/dz$ through the geometry of the look vector (see Figure 4-31).

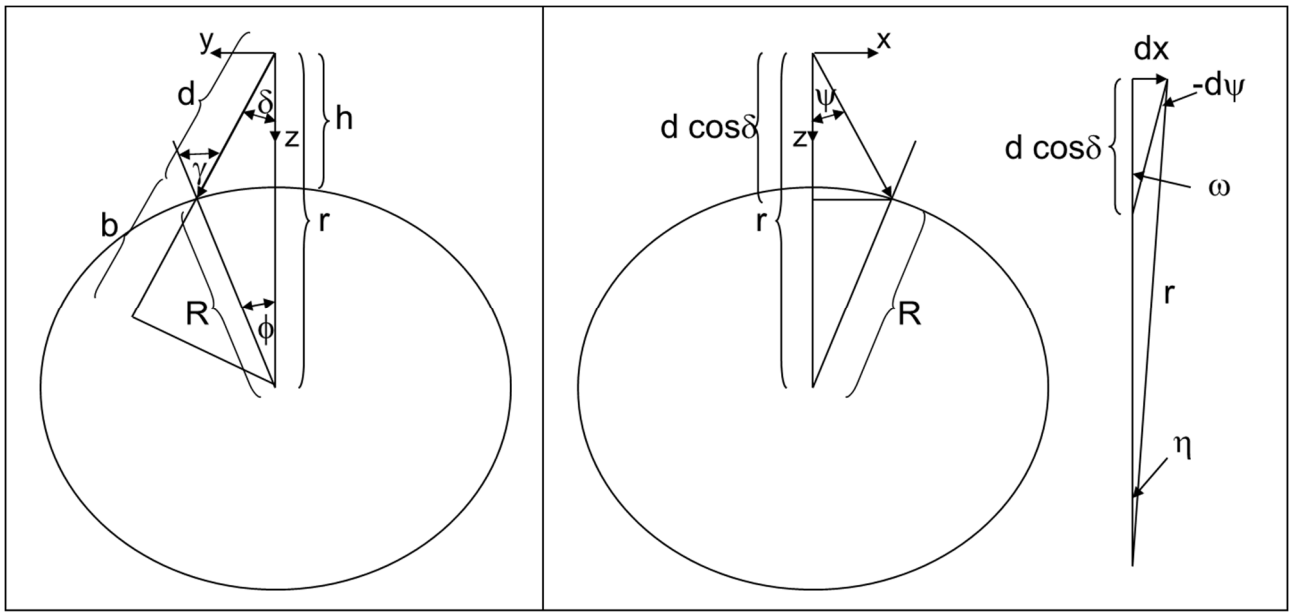

Figure 4-31. Look Vector Geometry

R – the radius of the Earth
r – the radius of the satellite position
h – the altitude of the satellite

d – the magnitude of the look vector (from satellite to target)
δ – the across-track angle of the look vector
ϕ – the Earth-centered angle between the satellite and the target
γ – the zenith angle of the look vector at the target
x,y,z – the coordinates of the satellite position in the OB system

We have

$$R \sin(\delta + \phi) = r \sin \delta \qquad (2.9)$$

Differentiating the equation (holding R and r constant) yields

$$R \cos(\delta + \phi)(d\delta + d\phi) = r \cos\delta \, d\delta \qquad (2.10)$$

Note that when $\delta + \phi = \gamma$, and $d\phi = -dy / r$, we have

$$\alpha = d\delta = (-b / (r\, d)) \, dy \qquad (2.11)$$

For the along-track direction, noting that d cosδ is the projection of the look vector onto the orbital radius vector, and referring to Figure 4-31:

A positive dx displacement causes a negative (backward) $d\psi$ so: $-d\psi = \omega - \eta$

The angle at the geocenter (η) is: $\eta = dx / r$

The angle ω is: $\omega = dx / d\cos\delta$

Substituting: $-d\psi = dx / d\cos\delta - dx / r$

Rearranging, we have

$$\beta = d\psi = (-(r - d\cos\delta) / (r\, d\cos\delta)) \, dx \qquad (2.12)$$

For the effect of altitude error, differentiate equation (2.9) with respect to δ and r (holding ϕ constant) and noting dr = -dz, we have

$$\alpha = d\delta = (\sin \delta / d) \, dz \qquad (2.13)$$

Note that the dx, dy, and dz in equations (2.11) through (2.13) are error terms, which are opposite in sign to the correction terms. With this in mind, we can replace the correction terms in equation (2.7) and (2.8) and rewrite the linearized observation equation as the following:

$$\alpha = (b / (r\, d)) \, dy - (\sin \delta / d) \, dz + d\theta_r \qquad (2.14)$$

$$\beta = ((r - d\cos\delta) / (rd\cos\delta)) \, dx - d\theta_p + \tan \delta \, d\theta_y \qquad (2.15)$$

where:

$$b = R\cos\gamma = \mathrm{sqrt}(R^2 - (r^2\sin^2\delta)) \quad (2.16)$$

$$d = r\cos\delta - b \quad (2.17)$$

This formulation does not account for the effects of applying the attitude correction in the ACS/body frame rather than the orbital frame. This is particularly significant in the case of off-nadir pointing. In the general case, applying the attitude correction in the ACS coordinate system leads to the following linearized observation equations:

$$\alpha = (b / (r\,d))\,dy - (\sin\delta / d)\,dz + M_{11}\,d\theta_r + M_{12}\,d\theta_p + M_{13}\,d\theta_y \quad (2.18)$$

$$\beta = ((r - d\cos\delta) / (rd\cos\delta))\,dx + (M_{31}\tan\delta - M_{21})\,d\theta_r + (M_{32}\tan\delta - M_{22})\,d\theta_p + (M_{33}\tan\delta - M_{23})\,d\theta_y \quad (2.19)$$

where:

M_{11}, M_{12}, M_{13}, M_{21}, M_{22}, M_{23}, M_{31}, M_{32}, M_{33} are the elements of the ACS to Orbital rotation matrix $\mathbf{M}_{ACS2ORB}$ at the time of the GCP observation. Thus, it is necessary to know the spacecraft roll-pitch-yaw corresponding to the GCP.

$$\mathbf{M}_{ACS2ORB} = \begin{bmatrix} \cos(p)\cos(y) & \sin(r)\sin(p)\cos(y) + \cos(r)\sin(y) & \sin(r)\sin(y) - \cos(r)\sin(p)\cos(y) \\ -\cos(p)\sin(y) & \cos(r)\cos(y) - \sin(r)\sin(p)\sin(y) & \cos(r)\sin(p)\sin(y) + \sin(r)\cos(y) \\ \sin(p) & -\sin(r)\cos(p) & \cos(r)\cos(p) \end{bmatrix}$$

Note that for nominal nadir viewing $M_{11} = M_{22} = M_{33} = 1$ and $M_{12} = M_{13} = M_{21} = M_{23} = M_{31} = M_{32} = 0$ and equations (2.18) and (2.19) reduce to equations (2.14) and (2.15).

Both linearized observation equations (2.18) and (2.19) include all three attitude correction terms. This has the effect of linking the along- and across-track observations in the OLI formulation, unlike the heritage implementation, which used separate along- and across-track solutions.

3. Weighted Least-Squares Solution

A weighted least-squares solution to the parameters is found using the following steps. The correction parameters in equations (2.18) and (2.19) can be expanded to include the correction to the change rates of the satellite attitude and position by defining

$$dx = dx_0 + dx_{dot}dt \quad \text{and} \quad d\theta_r = d\theta_{r0} + d\theta_{rdot}dt \quad (3.1)$$

Since both the coordinates of the GCP and the measurement of the apparent GCP location in the image contribute random errors in computing α and β, the covariance

matrix for the observation equations (2.18) and (2.19) should be the sum of the covariance matrix of $\mathbf{R}_{cp}$ in equation (1.6) and the covariance matrix of $\mathbf{R}_{ef}$ in equation (2.1), mapped through equations (1.7), (2.3), and (2.4).

Note that in the observation equations (2.14) / (2.15) and (2.18a) / (2.19a), α is only related to parameters dy, dz, and $d\theta_r$, and β is only related to dx, $d\theta_p$, and $d\theta_y$. The parameters are uncoupled in the two observations. In the simplified case where observational error of α and β are uncorrelated, the observation equations can be separated into two independent equations and solved individually. In the more general case of equations (2.18) and (2.19) the equations are coupled and must be solved together. This coupling is present in nadir-viewing scenes due to the yaw offsets introduced by yaw steering.

While it might be tempting to try to circumvent this complication by redefining the orbital coordinate system to be based on the Earth-rotation corrected ECEF velocity vector (thereby "yaw-steering" the orbital coordinate system), this would lead to a different set of complications in the application of the ephemeris corrections. In the baseline algorithm, we will adopt the general formulation of equations (2.18) and (2.19) and have adjusted the heritage separable least-squares solution formulation accordingly.

Proceeding with the integrated formulation, we define the parameter vector as the following:

$$\mathbf{X}' = \{d\theta_{r0}, d\theta_{p0}, d\theta_{y0}, dx_0, dy_0, dz_0, d\theta_{rdot}, d\theta_{pdot}, d\theta_{ydot}, dx_{dot}, dy_{dot}, dz_{dot}\} \qquad (3.2)$$

where ' means transpose of a vector or matrix. Then, the two observation equations can be written as follows:

$$\alpha = h_1 \mathbf{X} + \varepsilon_a \qquad (3.3)$$

$$\beta = h_2 \mathbf{X} + \varepsilon_b \qquad (3.4)$$

where:

$$h_1 = \{ M_{11}, M_{12}, M_{13}, 0.0, b/(d\ r), -\sin\delta/d, M_{11}\ dt, M_{12}\ dt, M_{13}\ dt, 0.0, b\ dt/(d\ r), -\sin\delta\ dt/d\} \qquad (3.5)$$

$$h_2 = \{(M_{31}\tan\delta - M_{21}), (M_{32}\tan\delta - M_{22}), (M_{33}\tan\delta - M_{23}), (r - d\cos\delta)/(r\ d\cos\delta), 0.0, 0.0, (M_{31}\tan\delta - M_{21})\ dt, (M_{32}\tan\delta - M_{22})\ dt, (M_{33}\tan\delta - M_{23})\ dt, (r - d\cos\delta)\ dt/(r\ d\cos\delta), 0.0, 0.0\} \qquad (3.6)$$

with M_{11}, M_{12}, M_{13}, M_{21}, M_{22}, M_{23}, M_{31}, M_{32}, M_{33} the elements of the ACS to Orbital rotation matrix $\mathbf{M}_{ACS2ORB}$ at the time of the GCP observation.

ε_a and ε_b are the random error of α and β, respectively. With all GCPs included, the along- and across-track observation equation can be written as follows:

$$\boldsymbol{A} = H_1\mathbf{X} + \varepsilon_A \quad (3.7)$$

$$\boldsymbol{B} = H_2\mathbf{X} + \varepsilon_B \quad (3.8)$$

and the integrated parameters can be solved by WLS estimation as follows:

$$\mathbf{X} = (H_1'WaH_1 + H_2'WbH_2)^{-1}(H_1'Wa\boldsymbol{A} + H_2'Wb\boldsymbol{B}) \quad (3.9)$$

where ***A*** and ***B*** are the observation vectors, composed of α and β for all the GCPs, respectively. H_1 and H_2 are corresponding coefficient matrix, with h_1 and h_2 as rows corresponding to each α and β, *Wa* and *Wb* are the diagonal weight matrix for ***A*** and ***B***, respectively, composed of inverse of the variance of each individual ε_a and ε_b.

4. Parameter Correlation and Covariance Estimation

One problem in this solution is the nearly linear correlation between parameter dx and $d\theta_p$ in the observation equation (3.7). The along-track orbit error and the pitch angle error have the very similar effect on β. The two parameters cannot be well separated in the solution without additional information – including both parameters in the observation equations results in a near-singular normal equation and therefore an unstable solution of the parameters. Similarly, high correlation exists between the cross-track position and the roll attitude errors in equation (3.6), and an ill-conditioned normal equation would result.

To correct the image, we do not have to distinguish between orbit position correction and attitude correction parameters. Letting either the orbit or the attitude correction parameters absorb the existing errors will correct the image in a similar manner. Therefore, we can choose to estimate either dx and dy or $d\theta_p$ and $d\theta_r$. This can be done by setting those coefficients in h_1 and h_2, corresponding to the unwanted parameters to zero.

One of the challenging tasks is to distinguish satellite attitude error from the orbit positional error. The purpose of precision correction estimation is not only to correct the image but also to extract information about the sensor alignment, which is reflected in the attitude correction parameters. To separate the ephemeris error from the attitude error as much as possible, we should first use the most precise ephemeris data available and correct systematic errors with available models. Second, we should use available a priori information, in addition to the observation, to cure the ill condition of the normal equation in statistical estimation.

Let the observation equation be:

$$\boldsymbol{Y} = H\boldsymbol{X} + \varepsilon;$$
$$E[\varepsilon] = 0, \quad Cov[\varepsilon] = s^2C \quad (4.1)$$

where $\boldsymbol{Y}$ is the measurement vector, $\boldsymbol{X}$ the parameter vector, H the coefficient matrix, ε the residual error vector, and s^2 is a covariance scaling factor

and the a priori information of the parameters are as follows:

$$\boldsymbol{X}\_ = \boldsymbol{X} + \varepsilon_x;$$
$$E[\varepsilon_x] = 0, \ \ Cov[\varepsilon_x] \ = q^2 C_x \qquad (4.2)$$

where $\boldsymbol{X}\_$ is the apriori parameter vector, ε_x is the residual vector, and q^2 is a covariance scaling factor; then the normal equation for the Best Linear Unbiased Estimate (BLUE) $\boldsymbol{X}\hat{}$ of the unknown parameter vector $\boldsymbol{X}$ is as follows:

$$((l/s^2)H'WH + (l/q^2)W_x)\boldsymbol{X}\hat{} = (l/s^2)H'W\boldsymbol{Y} + (l/q^2)W_x\boldsymbol{X}\_ \qquad (4.3)$$

where W and W_x are weight matrices.

$$W = C^{-1};$$

$$W_x = C_x^{-1} \qquad (4.4)$$

The covariance matrix of $\boldsymbol{X}\hat{}$ is as follows:

$$Cov[\boldsymbol{X}\hat{}] = ((l/s^2)H'WH + (l/q^2)W_x)^{-1} \qquad (4.5)$$

Usually, the Cov[ε] and Cov[ε_x] cannot be exactly known. In the case of GCP, for example, the position error involves many factors, such as base map error, and human marking error, etc. If there are unknown scale factors s^2 and q^2, we can still obtain the WLS estimate from the normal equation.

$$(H'WH + W_x)\boldsymbol{X}\hat{} = H'W\boldsymbol{Y} + W_x\boldsymbol{X}\_ \qquad (4.6)$$

In such case, the inverse of the normal matrix cannot be taken directly as the Cov[$\boldsymbol{X}\hat{}$]. Factor s^2 and q^2 should be estimated with appropriate variance component estimation from the residual of the solution of equation (4.6). The weighted residual square summation can be calculated as follows:

$$V'W\boldsymbol{V} = \boldsymbol{Y}'W\boldsymbol{Y} - 2\,\boldsymbol{X}\hat{}'M + \boldsymbol{X}\hat{}'\,N\boldsymbol{X}\hat{} \qquad (4.7)$$

$$V_x'W_x\boldsymbol{V_x} = \boldsymbol{X}\_'W_x\boldsymbol{X}\_ - 2\,\boldsymbol{X}\hat{}'W_x\boldsymbol{X}\_ + \boldsymbol{X}\hat{}'W_x\boldsymbol{X}\hat{} \qquad (4.8)$$

where:

$V = \boldsymbol{Y} - H\boldsymbol{X}\hat{}$ the measurement residual vector (4.9)

$\boldsymbol{V}_x = \boldsymbol{X}\_ - \boldsymbol{X}\hat{}$ the apriori parameter residual vector (4.10)

$$N = H'WH \quad (4.11)$$

$$M = H'W\mathbf{Y} \quad (4.12)$$

When the factors s^2 and q^2 are appropriately estimated, the weight matrix $\mathbf{W}$ and $\mathbf{W_x}$ should be correspondingly corrected by factors $1/s^2$ and $1/q^2$, respectively. Equation (4.6) should be resolved with the new weight matrices. In the new solution, information from the observation and the a priori information are appropriately combined and the $(H'\,WH + W_x)^{-1}$ is the Cov[$\mathbf{X\hat{}}$].

5. Weight Factor Estimation

One of the estimates of s^2 and q^2 is the Helmert type estimate. For the problem here, the equation for the estimate can be derived following Helmert's variance component analysis,

$$E\,s^2 + D\,q^2 = \mathbf{V'WV} \quad (5.1)$$

$$D\,s^2 + G\,q^2 = \mathbf{V_x'}W_x\mathbf{V_x} \quad (5.2)$$

where:

$$E = n - 2\,tr\{Q\,N\} + tr\{Q\,N\,Q\,N\} \quad (5.3)$$

$$G = m - 2\,tr\{Q\,W_x\} + tr\{Q\,W_x\,Q\,W_x\} \quad (5.4)$$

$$D = tr\{Q\,N\,Q\,Wx\} \quad (5.5)$$

$$Q = (H'\,W\,H + W_x)^{-1} \quad (5.6)$$

n = number of observations

m = number of parameters

$tr\{A\}$ indicates the trace of matrix A

Equation (5.1) and (5.2) do not guarantee positive solution of s^2 and q^2. In some cases, especially for small s^2 and q^2, noise can drive the solution negative. Another type of estimate, the iterative Maximum Likelihood Estimate (MLH), guarantees a positive solution, though the estimate s^2 and q^2 may not be statistically unbiased. The MLH solution is obtained by iteratively solving equation (4.6) and

$$s^2 = \mathbf{V'WV} / n \quad (5.7)$$

$$q^2 = \mathbf{V_x'}W_x\mathbf{V_x} / m \quad (5.8)$$

$$W = W / s^2 \quad (5.9)$$

$$W_x = W_x / q^2 \quad (5.10)$$

until s^2 and q^2 converge.

The solution above provides the estimate of the corrections to the ephemeris and attitude data, as well as to their covariance matrix. The covariance information can be used as a measure of precision for assessing the alignment errors of the sensor system. It can also be propagated to any pixel in the scene to evaluate the pixel location error after the precision correction.

6. Covariance Propagation

Given the sample time and across-track look angle of a pixel, the coefficients h_1 and h_2 can be calculated for α and β according to equation (3.6) and (3.7). The variance of α and β are then calculated as:

$$\sigma_\alpha^2 = h_1 \, \mathrm{Cov}[\mathbf{X^\wedge}] h_1' \quad (6.1)$$

$$\sigma_\beta^2 = h_2 \, \mathrm{Cov}[\mathbf{X^\wedge}] h_2' \quad (6.2)$$

These are the variance of the pixel location in sample and line directions due to the uncertainty of the estimated precision-correction parameters. They are in angles, but can be easily converted into IFOV according to the sensor system specifications.

7. Outlier Detection

Outlier detection for the precision-correction solutions seeks to identify GCPs that are likely to be in error due to miscorrelation. This is done by analyzing the GCP residuals, taking into account the relative importance of the GCP as reflected in the precision solution normal equation matrix.

Definitions:

A = matrix of coefficients (partial derivatives) relating parameters to observations
$\underline{\theta}$ = parameter vector
$\underline{X}$ = observation vector
$\underline{V}$ = residual vector
C = observation covariance matrix
n = the number of observations
p = the number of parameters
A is n x p, $\underline{\theta}$ is p x 1, $\underline{X}$ and $\underline{V}$ are n x 1, and **C** is n x n

Observation Equation:

$$\mathbf{A}\underline{\theta} = \underline{X} - \underline{V} \quad (7.1)$$

$$\underline{X} = \underline{X}_{true} + \underline{E} \qquad \text{where E = error vector} \sim G(\underline{0}, \mathbf{C}) \quad (7.2)$$

$\mathbf{A}\underline{\theta}_{true} = \underline{X}_{true}$ where $\underline{\theta}_{true}$ is the "true" parameter vector (7.3)

$\mathbf{A}\underline{\theta} = \underline{X}_{true} + \underline{E} - \underline{V}$ (7.4)
so $\underline{V} = \underline{E}$ if $\underline{\theta} = \underline{\theta}_{true}$

Minimum Variance Parameter Estimate:
$\underline{\theta}' = [\mathbf{A}^T\mathbf{C}^{-1}\mathbf{A}]^{-1}\mathbf{A}^T\mathbf{C}^{-1}\underline{X}$ (7.5)

Estimated Residual Error:
$\underline{V}' = \underline{X} - \mathbf{A}[\mathbf{A}^T\mathbf{C}^{-1}\mathbf{A}]^{-1}\mathbf{A}^T\mathbf{C}^{-1}\underline{X}$ (7.6)

Define Projection matrix **P**:
$\mathbf{P} = \mathbf{A}[\mathbf{A}^T\mathbf{C}^{-1}\mathbf{A}]^{-1}\mathbf{A}^T\mathbf{C}^{-1}$ (7.7)
This matrix projects the observation vector into the parameter subspace (the column space of **A**). This projection is only orthogonal if **C** has the special structure described below.

Substituting:
$\underline{V}' = \underline{X} - \mathbf{P}\underline{X} = [\mathbf{I} - \mathbf{P}]\underline{X}$ (7.8)
$[\mathbf{I} - \mathbf{P}]$ projects $\underline{X}$ into the parameter null space.

Looking at the Error Estimate $\underline{V}'$:
$\underline{V}' = [\mathbf{I} - \mathbf{P}]\underline{X} = [\mathbf{I} - \mathbf{P}][\underline{X}_{true} + \underline{E}] = [\mathbf{I} - \mathbf{P}]\underline{X}_{true} + [\mathbf{I} - \mathbf{P}]\underline{E}$ (7.9)

but $[\mathbf{I} - \mathbf{P}]\underline{X}_{true} = \underline{0}$ since $\underline{X}_{true}$ lies entirely within the parameter subspace.

so $\underline{V}' = [\mathbf{I} - \mathbf{P}]\underline{E} = \underline{E} - \mathbf{P}\underline{E}$ (7.10)

Here are some comments about $\underline{V}'$ and $\underline{E}$:
- For a given precision solution, the elements of $\underline{E}$ are not random variables; they are realizations of random variables.
- $\underline{V}'$ is an estimate of the actual (realized) error $\underline{E}$, which includes an estimation error equal to $\mathbf{P}\underline{E}$.
- We cannot exactly recover $\underline{E}$ from $[\mathbf{I} - \mathbf{P}]^{-1}\underline{V}'$ because $[\mathbf{I} - \mathbf{P}]$ is singular (it is an n x n matrix of rank n-p).
- We can attempt to predict how accurate our estimate ($\underline{V}'$) of $\underline{E}$ is likely to be by looking at the estimation error $\underline{R} = \mathbf{P}\underline{E}$.
- Since we want the predicted accuracy to apply in general, we treat $\underline{R}$ as a random vector, which is a function of another random vector $\underline{E}$.

Expected Value: $E[\underline{R}] = E[\mathbf{P}\underline{E}] = \mathbf{P}\,E[\underline{E}] = \mathbf{P}\,\underline{0} = \underline{0}$ (7.11)
Variance: $E[\underline{R}\,\underline{R}^T] = E[\mathbf{P}\,\underline{E}\,\underline{E}^T\,\mathbf{P}^T] = \mathbf{P}\,E[\underline{E}\,\underline{E}^T]\,\mathbf{P}^T = \mathbf{P}\,\mathbf{C}\,\mathbf{P}^T$ (7.12)

Special Structure of Observation Covariance Matrix for Precision Correction:
$\mathbf{C} = \sigma^2\mathbf{I}$ (7.13)
since the observation errors are realizations of independent and identically distributed zero mean Gaussian random variables with variance σ^2.

Substituting (7.13) into equation (7.7) for **P** yields:

$$\mathbf{P} = \mathbf{A}[(1/\sigma^2)\mathbf{A}^T\mathbf{I}\mathbf{A}]^{-1}\mathbf{A}^T\mathbf{I}(1/\sigma^2) = \mathbf{A}\sigma^2[\mathbf{A}^T\mathbf{A}]^{-1}\mathbf{A}^T(1/\sigma^2) = \mathbf{A}[\mathbf{A}^T\mathbf{A}]^{-1}\mathbf{A}^T \quad (7.14)$$

And the equation for the variance of $\underline{R}$:

$$E[\ \underline{R}\ \underline{R}^T\ ] = \sigma^2\ \mathbf{P}\ \mathbf{I}\ \mathbf{P}^T = \sigma^2\ \mathbf{P} \quad (7.15)$$

noting that $\mathbf{P}^T = \mathbf{P}$ and $\mathbf{P}\ \mathbf{P} = \mathbf{P}$

so $\underline{R} \sim G(\ \underline{0},\ \sigma^2\ \mathbf{P}\ )$

For a particular component of $\underline{R}$ r_i:

$$E[\ r_i\ ] = 0 \quad (7.16)$$

$$E[\ r_i^2\ ] = \sigma^2\ p_{ii} \quad (7.17)$$

Where p_{ii} is the i^{th} diagonal component of **P**

Looking at the equation for **P**, we see that:

$$p_{ii} = \underline{A}_i^T\ [\mathbf{A}^T\mathbf{A}]^{-1}\ \underline{A}_i \quad (7.18)$$

Where $\underline{A}_i^T$ is the i^{th} row of **A**

Considering a particular component of the Residual Error Vector $\underline{V}$':

$$v_i = e_i - r_i \quad (7.19)$$

Where e_i is the corresponding component of the observation error vector

so v_i is an unbiased estimate of e_i with variance $\sigma^2\ p_{ii}$

If we knew what e_i was, we could test it against a probability threshold derived from its standard deviation, σ, to determine if it is likely to be an outlier. Instead of e_i, we have v_i, which includes the additional error term r_i. Including the additional estimation error in the threshold computation leads to the following:

$$\sigma_v^2 = \sigma^2 + \sigma^2\ p_{ii} \quad (7.20)$$

Where σ^2 is the term due to the actual error variance and $\sigma^2\ p_{ii}$ is the term due to the estimation error variance.

This may seem like cheating since e_i and r_i are not independent for a given realization.

$$E[\ v_i^2\ ] = E[\ (e_i - r_i)^2\ ] = E[\ e_i^2 - 2e_ir_i + r_i^2\ ] \qquad \text{and } r_i = \Sigma_j\ p_{ij}\ e_j$$

$$E[\ v_i^2\ ] = \sigma^2\ (1 - p_{ii}) \quad (7.21)$$

It is tempting to use $v_i\ /\ (1 - p_{ii})^{1/2}$ for e_i in the outlier test (or, equivalently, to test v_i against a threshold based on $\sigma^2\ (1 - p_{ii})$), but this becomes dangerous as p_{ii} approaches 1. The factor p_{ii} can be interpreted as a measure of the uniqueness of, or as the information content of, the i^{th} observation. As p_{ii} approaches 1, the i^{th} observation lies almost entirely within the parameter subspace, which implies that it is providing information to the solution that the other observations do not. Note that such "influential" observations can be identified from the structure of the coefficient matrix, **A**, without reference to the observation residuals. Attempting to use $1/(1 - p_{ii})^{1/2}$ to rescale the residual v_i to better approximate e_i will, in a sense, punish this observation for being important. Instead, we view p_{ii} as a measure of how poor an estimate of the actual error, e_i, the residual, v_i, is and ignore the fact that v_i will tend to be an underestimate of e_i. We

therefore use σ_v^2 (= σ^2 (1 + p_{ii}) as shown above) to construct the outlier detection threshold.

One remaining problem is that we do not know exactly what σ^2 is and must estimate it from the observation residuals. This is done by scaling the a priori observation variance using the variance of unit weight that was computed in the precision solution. The fact that we are using an estimated variance to establish our outlier detection threshold modifies the algorithm in two ways: 1) we compensate for the fact that removing a point as an outlier will alter the computation of the variance of unit weight by removing one residual and reducing the number of degrees of freedom; and 2) we base the detection threshold computation on student's t-distribution rather than the Gaussian distribution.

The variance of unit weight is computed as follows:

$$var_0 = \underline{V}^T\mathbf{C}^{-1}\underline{V}/(n - p) = \underline{V}^T\underline{V}/\sigma_0^2(n - p) = \Sigma v_j^2/\sigma_0^2(n - p) \quad (7.22)$$

Where: n = number of observations,
p = number of parameters, and
σ_0^2 is the a priori variance.

The estimated variance is as follows:

$$var = var_0\ \sigma_0^2 = \Sigma v_j^2/(n - p) \quad (7.23)$$

Removing the k^{th} observation makes this:

$$var_k = (\Sigma v_j^2 - v_k^2)/(n - 1 - p) = (n - p)/(n - p - 1) * (\Sigma v_j^2 - v_k^2)/(n - p)$$

$$var_k = (n - p)/(n - p - 1) * var - v_k^2/(n - 1 - p) \quad (7.24)$$

To normalize the k^{th} residual, we divide it by the estimated standard deviation $\sigma' = (var)^{1/2}$:

$$w_k = v_k / \sigma' \quad (7.25)$$

We can rescale this normalized residual to reflect the removal of this observation from the variance estimate without having to actually compute a new variance:

$$w_k' = v_k / \sigma_k' = w_k\ \sigma'/\sigma_k' = w_k\ (var/var_k)^{1/2}$$

$$var/var_k = 1 / [(n - p)/(n - p - 1) - v_k^2/var\ (n - p - 1)] = (n - p - 1)/(n - p - v_k^2/var)$$

$$var/var_k = (n - p - 1)/(n - p - w_k^2)$$

noting that $v_k^2/var = w_k^2$

$$w_k' = w_k\ [(n - p - 1)/(n - p - w_k^2)]^{1/2} \quad (7.26)$$

Finally, we include the (1 + p_{kk}) factor discussed above and our normalized and reweighted residual becomes:

$$w_k' = w_k\ [(n - p - 1)/(1 + p_{kk})(n - p - w_k^2)]^{1/2} \quad (7.27)$$

where: $w_k = v_k / \sigma'$

This normalized and reweighted residual is compared against a probability threshold computed using Student's t-distribution with (n - p) degrees of freedom.

LOS Correction Procedure Overview

The precision-correction procedure developed mathematically above is implemented as an iterative solution to account for the non-linearity of the observation equations presented in (2.5) and (2.6) above. Each step in the iteration solves the linearized correction problem using equation (3.10) above, using the current correction estimates, to compute incremental corrections for the current iteration. These corrections are used to update the current estimates, and the iteration continues until the incremental corrections are smaller than some threshold (or the iteration limit is exceeded).

An additional layer of iteration is introduced by the need to perform outlier filtering on the input GCP data. This, the procedure includes two levels of iteration: 1) use the current active set of GCPs to perform the iterative weighted least-squares solution (the linearization iteration); 2) filter the resulting GCP residuals for outliers, remove those exceeding the specified tolerance, and iterate the weighted least-squares procedure with the new (reduced) active set until no new outliers are found.

The LOS correction procedure can be viewed as a five-phase process in which the third and fourth phases are nested:

a) Phase 1 - Load the necessary data and initialize the solution procedure.
b) Phase 2 - Load and initialize the GCPs. For each GCP, use the geometric grid to compute the input space (L1R) location and time of observation. Interpolate the spacecraft position, velocity, and attitude at the time of observation.
c) Phase 3 - Use the current active set of GCPs to form and solve the linearized weighted least-squares equation. Use the computed corrections to update the current estimates. Iterate the linearized solution procedure until the incremental corrections are below the convergence threshold. Compute and write residuals for each iteration (the initial pre-correction and final iteration residuals are both used in geodetic accuracy assessment). Note that the residuals file is reinitialized at the beginning of each phase 4 loop so that the output residual file will reflect only the final pass through the outlier detection loop (phase 4).
d) Phase 4 - Run the outlier detection and removal iteration loop using the results of the iterative weighted least-squares solution procedure (phase 3) by testing the resulting residuals for outliers. Remove any newly detected outliers from the active GCP list and recompute the phase 3 solution with the reduced GCP set. Continue to iterate until no new outliers are detected.
e) Phase 5 - Write the precision solution file to document the result, update the LOS model, and, if requested, convert the attitude corrections to OLI alignment angles and write the resulting alignment calibration information to the characterization database.

Figure 4-32 shows a block diagram of the LOS correction procedure in which the individual process steps are identified by phase using the codes P1 through P5.

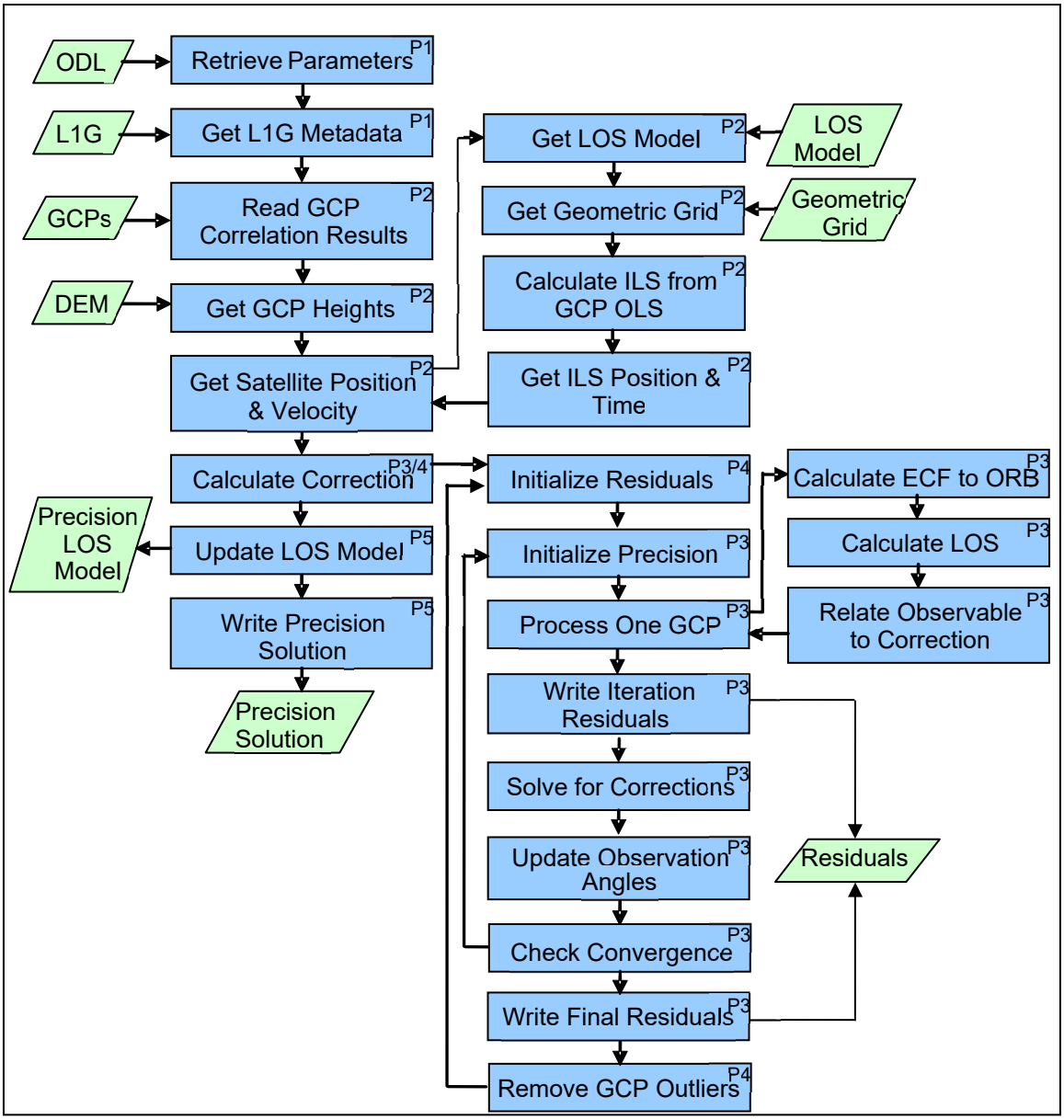

Figure 4-32. LOS Correction Algorithm Block Diagram

The following text describes the individual process steps in the LOS correction procedure.

4.2.3.7 Detailed Procedure

Inputs to the algorithm include an ODL parameter file, an ASCII GCP measurement file created by the GCP correlation algorithm, the L1G image used to measure the GCPs, the OLI LOS model used to create the L1G image, the LOS projection grid file used to create the L1G image, the CPF used to create the L1G image, and the DEM file (if any) used to terrain correct the L1G image. Note that only the L1G image metadata is used, not the imagery itself. The outputs are an updated (precision corrected) OLI LOS model

file, an ASCII report file containing a standard header that identifies the dataset analyzed, and the results of the precision-correction solution, and an ASCII file containing the computed GCP residual errors at each iteration of the solution. The GCP residuals for the first and last iteration are subsequently used by the geodetic accuracy characterization algorithm.

Get Precision Parameters
Get the precision-correction algorithm parameters.

Read Precision Parameters
Read all of the parameters that precision requires from ODL and CPF files.

Get Position
Find the satellite position, velocity, attitude, and reference time for each GCP.

Add Position
Add the position to the ground control point structure and assign the reference time to the time structure, using the following steps:

1. From the 1G line and sample, find the latitude and longitude, and use the DEM to find the height at that line/sample; then, transform to Earth-fixed.
2. From the L1G line and sample, use the geometric grid and the ols2ils routine (reference the LOS Projection Algorithm) to compute the corresponding input space line and sample. Note that this computation includes the DEM height interpolated in step 1 above.
3. From the input space line and sample, calculate the reference year, day, and seconds, satellite position and velocity, and spacecraft attitude (roll-pitch-yaw).
4. Calculate the transformation matrix from Earth-fixed to orbit-oriented.
5. Calculate the line-of-sight.

Calculate Position
Find the satellite position, velocity, attitude, and time, using the forward model. This sub-algorithm invokes oli_findtime to get the time, oli_findatt to get the attitude, and l8_movesat to compute the position and velocity. The OLI LOS Projection Algorithm describes these sub-algorithms.

Get Latitude/Longitude
Find the latitude/longitude given the L1G line/sample.

1. Find the first-order rotation coefficients if there is a rotation.
2. Find the output projection coordinate of pixel.
3. Call the projtran routine (see the LOS Projection Algorithm for details) to convert projection X/Y coordinates to the corresponding latitude/longitude.
4. Access the DEM to interpolate the height at the L1G line/sample coordinates. Note that this is a departure from the heritage approach and is a consequence of using terrain-corrected mensuration images.

5. Convert the latitude, longitude, and height into Cartesian ECEF coordinates (x,y,z), as described below.

Geodetic to Cartesian
Convert geodetic coordinates (lat, lon, height) into Cartesian coordinates (x, y, z), as described in the LOS Projection Algorithm and reiterated below. Input latitude and longitude are in radians; height, semi-major axis, and output Cartesian position vector are in meters; flattening is a dimensionless number.

$b = a (1 - f)$
$e^2 = 1 - b^2 / a^2$
$N = a / (1 - e^2 \sin^2(\varphi))^{1/2}$
$X = (N + h) \cos(\varphi) \cos(\lambda)$
$Y = (N + h) \cos(\varphi) \sin(\lambda)$
$Z = (N (1-e^2) + h) \sin(\varphi)$

where:

X, Y, Z	-	ECEF coordinates
φ, λ, h	-	Geodetic coordinates (lat ϕ, long λ, height h)
N	-	Ellipsoid radius of curvature in the prime vertical
f	-	Ellipsoid flattening (f = 1 - b/a)
e^2	-	Ellipsoid eccentricity squared
a, b	-	Ellipsoid semi-major and semi-minor axes

Calculate Line-of-Sight
Calculate the line-of-sight angles from the satellite position to the ground point from their position coordinates.

For x, y, and z, assign: ecf_look = pixpos - satpos.

Perform matrix multiplication to transform Earth-fixed look vector to orbit-oriented look vector (see the Earth-Fixed to Orbit-Oriented sub-algorithm below for the construction of the Tecf2oo transformation matrix):

$$[\text{Tecf2oo}]_{3x3} [\text{ecf\_look}]_{3x1} = [\text{oo\_look}]_{3x1}$$

Compute the along- and across-track angles:

psi = arctan(oo_look[0] / oo_look[2])
delta = arctan(oo_look[1] / oo_look[2])

Calculate Correction
Solve for the attitude and/or ephemeris correction using the Ground Control Points.

1. Initialize the correction parameter structure.
2. Allocate memory for the residuals structure.
3. Begin the outlier detection and rejection iteration loop.

4. Prepare the residual file to be written to.
5. Reset the GCP information to its original state.
6. Initialize the weight factor for observation and a priori parameters.
7. Iterate the precision correction solution process.
   a) Initialize the normal equations.
   b) For each GCP, compute the observables (α and β), relate them to the correction parameters, and then form the normal equation to accumulate.
   c) Accumulate the normal equations by adding up information from each GCP.
   d) compute diff_time = gcps[gcp_num].time - ref_time[2]
   e) Write the residual information for this iteration. This will be done for each iteration. The structure get_residuals must be filled before writing to this file. We store the RMS residuals for the first and last iteration as solution quality metrics.
   f) Solve the Normal equations. Solve for the corrections from the normal equation using the Weighted Least-Square sub-algorithm.
   g) If the parameter flag is 4 (weight factor estimation option):
      1. Estimate the variance factor with the Minimum Norm Quadratic Unbiased Estimate (MINQUE).
      2. If the MINQUE solution is obtained, compute the residual square sum.
      3. If the MINQUE solution failed, try the Maximum Likelihood Estimate (MLHE) solution.
      4. If MLHE fails, the solution cannot be obtained.
      5. Calculate the posteriori standard error.

      Else If the parameter flag is not 4 (no a priori weight factor estimation is used):
      1. Compute the residual square sum.
      2. Calculate the posteriori standard error.
   h) Update the total correction estimate.
   i) Update the observable and orbit state for each GCP.
   j) If the sum of the absolute values of the elements of the across-track and along-track solution vectors are greater than 1 and the number of iterations is less than max iterations, iterate again; otherwise, end iteration.
8. Calculate the residual in alpha and beta for each GCP.
9. Check the residuals for new outliers; if any are found, continue the outlier iteration loop from step 3 above.
10. Extract the final solution and update the correction parameters.

Write Residuals

Write out the along- and across-track residual components for each GCP to the residual file.

For each GCP:

Rescale the residuals to meters.
Compute the projection space value of the residuals.
Copy the information to the residual structure.
Write out the residual information.

Get Ground
Calculate the projection (x/y) residual values in meters from the Earth orbit delta and psi residual values.

1. Calculate the Earth Centered Fixed (ECF) to Orbit Oriented (OO) transformation system and transpose the matrix to get the OO to ECF matrix.
2. Given the satellite position and correction terms, calculate a new look vector.
3. Transform the vector from OO to ECF.
4. Convert the ECF latitude and longitude to projection in meters.
5. Convert the true latitude and longitude to projection in meters.
6. Subtract the projection and assign to residual.

Earth-fixed to Orbit-oriented
Generate the transformation matrix from the Earth-fixed Cartesian system to the OO Cartesian system, as described in the Ancillary Data Preprocessing Algorithm Section 4.1.4 and reiterated below. Note that the ECEF velocity vector is really the ECI velocity vector rotated into the ECEF coordinate system (i.e., it is still an inertial velocity) and does not include the relative Earth rotation velocity. This is done so that the ECEF velocity vector remains parallel to the attitude control reference X axis, which is defined in ECI coordinates.

The relationship between the orbital and Earth-Centered coordinate systems is based on the spacecraft's instantaneous ECEF position and velocity vectors. The rotation matrix to convert from orbital to ECEF can be constructed by forming the orbital coordinate system axes in ECEF coordinates:

$$\vec{n} = -\frac{\vec{p}}{\left\|\vec{p}\right\|}$$

$$\vec{h} = \frac{\left(\vec{n} \times \vec{v}\right)}{\left\|\vec{n} \times \vec{v}\right\|}$$

$$\vec{cv} = \vec{h} \times \vec{n}$$

$$\left[\text{ORB2EC}\right] = \left[\vec{cv} \quad \vec{h} \quad \vec{n}\right]$$

where:

p = spacecraft position vector in ECEF

v = spacecraft velocity vector in ECEF
n = nadir vector direction
h = negative of angular momentum vector direction
cv = circular velocity vector direction
[ORB2ECEF] = rotation matrix from orbital to ECEF

The transformation from orbital to ECEF coordinates is the inverse of the ECEF to orbital transformation matrix. Since the ECEF to orbital matrix is orthogonal, the inverse is also equal to the transpose of the matrix.

$$[ORB2ECEF] = [ECEF2ORB]^{-1} = [ECEF2ORB]^{T}$$

Detect Outliers

Detect GCP outliers using the residuals and normal equations. Given a tolerance value, outliers are removed within the dataset until all values deemed as "non-outliers" or "valid" fall inside the confidence interval of a T-distribution. The tolerance, or associated confidence interval, is specified per run and usually lies between 0.9-0.99. The default value is 0.95. The number of degrees of freedom of the dataset is equal to the number of valid data points minus one. The steps involved in this outlier procedure are as follows:

1. Calculate the standard deviation of all valid points in the dataset.
2. Loop on "valid" data points until no outliers are found.
   a) Find the two-tailed T-distribution (*T*) value for the current degree of freedom and confidence level specified α.
   b) Calculate the largest deviation allowable for the specified degree of freedom and α. This is not scaled by σ since the residuals themselves are normalized by σ in step c below.
      $\Delta = T$
   c) For each data point, compute the along- and across-track weight factors using equation (7.18) above and the normalized and weighted along- and across-track residuals using equation (7.27) above.
   d) Find the data point with the largest normalized and weighted residual.
   e) If the maximum residual value found in step d is less than Δ, then exit.
   f) If the value found in step d is greater than Δ, then flag the data point as an outlier and calculate the standard deviation of the new set of "valid" data points.

Get Correction

Extract the estimated correction parameters and their covariance matrix from the Weighted Least-Square solution, and update the correction parameter structure.

1. Record the reference time for the correction.
2. Extract the satellite position corrections.
3. Extract the satellite velocity corrections.
4. Extract the satellite attitude angle corrections.
5. Extract the satellite attitude angle rate corrections.

6. Record the covariance matrix for the correction parameters.

Reset Observations
Reset the satellite state vector and the look angles corresponding to each GCP to their original values. This resets the inputs for the next iteration of the outlier loop.

Initialize Precision
Initialize the normal matrix for the attitude and ephemeris correction estimate by least-square solutions.

1. Initialize the observational and a priori part of the normal equation, obs_mx, obs_rgt, apr_corr, apr_wgt_par, to zero or almost zero.
2. if param_flag = both or input weights are provided, estimate all corrections.
3. Form the a priori normal matrix for the parameters.
4. Form the a priori right-side term for the parameters.
5. Subtract the current net correction (Yb) terms from the right-hand side to restrain the magnitude of the net correction.
6. if param_flag = eph_yaw, estimate the orbit corrections:
   a) set zero a priori mean for roll
   b) set zero a priori mean for roll dot
   c) set huge a priori weight for roll
   d) set huge a priori weight for roll dot
   e) set zero a priori mean for pitch
   f) set zero a priori mean for pitch dot
   g) set huge a priori weight for pitch
   h) set huge a priori weight for pitch dot
7. if param_flag = att_orb, estimate attitude corrections:
   a) set zero a priori mean for dy
   b) set zero a priori mean for dy dot
   c) set huge a priori weight for dy
   d) set huge a priori weight for dy dot
   e) set zero a priori mean for dx
   f) set zero a priori mean for dx dot
   g) set huge a priori weight for dx
   h) set huge a priori weight for dx dot
8. if time_flag = FALSE
   a) Block out the rate terms by setting a huge weight for zero apriori mean.
9. Initialize the number of observations.
10. Initialize the weighted residual square summation.

Process One GCP
Update the normal equations of the least-square problem for the correction solution by adding one GCP.

Calculate the transformation matrix from ECF to Orbit system
Calculate the line-of-sight angles for GCP

Note: The look vectors here should be in the Orbit reference system.

If the line-of-sight angle for the pixel P_i is from the forward model (in the spacecraft-fixed system), then it should be transformed into the Orbit reference system (through matrix A(roll, pitch, yaw)) first before the observable alpha and beta can be formed.

Compute the observable alpha and beta
If not an outlier:
Relate the observable to correction parameters
Update the weighted square summation of observation
Accumulate the normal equation contribution for alpha
Accumulate the normal equation contribution for beta

Partial
Compose the partial coefficients matrix of the observation equation, given the angle delta for one GCP.

Partial Attitude
Compose the partial coefficients matrix of the observation equation for param_flag = "att_orb," estimating attitude plus height corrections.

Calculate the constants needed for the partial derivative (H) matrix calculation, such as sin(delta), cos(delta), and satellite radius.
The side perpendicular to the look vector = satellite_radius * sindelta
Compose the H matrix by finding:
alpha w.r.t roll, microradian
alpha w.r.t pitch, microradian
alpha w.r.t yaw, microradian
alpha w.r.t. dz, meter scaled to microradian
beta w.r.t roll, microradian
beta w.r.t. pitch, microradian
beta w.r.t. yaw, microradian

Partial Ephemeris
Compose the partial coefficients matrix of the observation equation for param_flag = "eph_yaw," estimating ephemeris plus yaw corrections.

Calculate the constants needed for H calculation by assigning sin(delta), cos(delta), and satellite radius.
Compose the H matrix by finding:
alpha w.r.t. dy, meter scaled to microradian
alpha w.r.t. dz, meter scaled to microradian
alpha w.r.t. yaw, microradian
beta w.r.t. dx, meter scaled to microradian
beta w.r.t. yaw, microradian

Partial All
Compose the partial coefficients matrix of the observation equation for param_flag = "both" or "weight," estimating both attitude and ephemeris corrections. Note that this is the normal case.

Calculate the constants needed for H calculation sin(delta), cos(delta), and satellite radius (see equations (3.6) and (3.7) above).
Compose the H matrix:
alpha w.r.t. roll, microradian
alpha w.r.t. pitch, microradian
alpha w.r.t. yaw, microradian
alpha w.r.t. dy, meter scaled to microradian
alpha w.r.t. dz, meter scaled to microradian
beta w.r.t. dx, meter scaled to microradian
beta w.r.t. roll, microradian
beta w.r.t. pitch, microradian
beta w.r.t. yaw, microradian

Accumulate Normal Equation
Accumulate the normal equation of the least-square problem by adding one observation.

Update the n x n normal matrix by accumulating:
H_transpose * wo * H

Update the n x 1 right-hand-side array of the normal equation by adding:
H_transpose * wo * obs

where:
H is the matrix of partial derivatives
wo is the observation weight
obs is the observation value

Weighted Least Square
Solve the weighted least-square problem with nxn normal matrix.

Form the normal equation for the Weighted Least-Square (WLS) problem, including any weight factors:
A[i][j] = weight_factor_for_observation * normal_matrix_for_observation[i][j]
Augment the diagonal terms using the apriori observations:
A[i][i] += weight_factor_for_apriori * normal_matrix_for_apriori[i]

Form the constant vector, including both observations and apriori contributions:
L[i] = weight_factor_for_observation * observation_rhs[i]
+ weight_factor_for_apriori * apr_corr[i]

Solve the equation:
solution = sol_Ya = A^{-1} L

Note that the inverted normal equation matrix (A^{-1}) is returned along with the solution so that it can be used to construct the solution aposteriori covariance matrix.

MINQUE
Estimate the variance factor with MINQUE.

let:
wght_rss_obs = weighted residual square for observation
cov_mx = Inverse of the WLS problem normal matrix
obs_mx = the observation part of the normal matrix
apr_wgt_par = the a priori weights loaded into a diagonal weight matrix
wgt_fact_obs = the estimated variance factor for the observation
wgt_fact_apr = the estimated variance factor for the a priori variance

compute the weighted residual square for the observation (rss_obs)

compute the weighted residual square for the a priori parameters (rss_apr)

Allocate memory for arrays

compute the trace coefficients for the weight estimate equation
cc2 = cov_mx * apr_wgt_par
cc1 = cov_mx * obs_mx
s1 = ngcp - 2tr[cc1] + tr[cc1 * cc1] ref. equation (5.3)
s2 = n_aprior - 2tr[cc2] + tr[cc2 * cc2] ref. equation (5.4)
s12 = tr[cc1 * cc2] ref. equation (5.5)

solve for the weight factors:
ss1 = s1 * s2 - s12 * s12
wgt_fact_obs = (rss_obs * s2 - rss_apr * s12) / ss1
wgt_fact_apr = (rss_apr * s1 - rss_obs * s12) / ss1

If wgt_fact_obs and wgt_fact_apr are less than 0.0--return, minque failed.
Run WLS where the scale factor for the weight of observation and a priori are 1/wgt_fact_obs and 1/wgt_fact_apr, respectively.
If WLS fails, return minque with failed status.
If WLS returns a non-error value, assign wght_rss_obs.

Residual Square Sum
Compute the residual square sum by adding the dot product of:
sol_Ya^T * obs_mx * sol_Ya - 2 * obs_rgt^T * sol_Ya
where:
sol_Ya is the weighted least-squares solution vector

obs_mx is the normal equation matrix
obs_rgt is the right hand side vector of the normal equations

to the observation square sum (post_sig).

MLHE
Estimate the variance factor with MLHE.

Initialize the weight factor to zero.
Iterate the estimation of the weight factors.
Compute the weighted residual square for the observation and for the apriori parameters.
Compute the weight factor estimate.
Compute the weight factor difference for this iteration.
Solve the new WLS solution with the new weight factors.
Compute the final variance factor estimate.

New Observation Angle
Update the satellite state vector and the look angles corresponding to each GCP, according to the correction parameters, for the purpose of iteration.

Extract the orbit and attitude correction parameters from the solution vectors.

Orbit corrections:
dorbit[0] = sol_Ya[3]
dorbit[1] = sol_Ya[4]
dorbit[2] = sol_Ya[5]
orbit_rate[0] = sol_Ya[9]
orbit_rate[1] = sol_Ya[10]
orbit_rate[2] = sol_Ya[11]

Attitude corrections:
datt[0] = sol_Ya[0]
datt[1] = sol_Ya[1]
datt[2] = sol_Ya[2]
att_rate[0] = sol_Ya[6]
att_rate[1] = sol_Ya[7]
att_rate[2] = sol_Ya[8]

For each GCP
Calculate the orbit perturbation and update the orbit state vector
Calculate the attitude perturbation and update the look angles

Update Ephemeris
Calculate the orbit position change and updates the ephemeris data in the Earth-Fixed system.

Construct the ECF to orbital transformation T_{ef2oo} from the input position and velocity vectors (using xxx_earth2orbit).

Take the transpose of (the orthogonal matrix) T_{ef2oo} to find the inverse T_{oo2ef}.

Transform the input orbital position and velocity corrections to ECF using T_{oo2ef}.

Update the input ECF position and velocity by adding the transformed position and velocity corrections.

Calculate New Look Angles

Calculate the new look angles by adding the attitude angle perturbation. The heritage implementation was modified as described below to account for applying the attitude corrections in the ACS rather than the orbital coordinate system.

Convert the units of the attitude corrections (to radians).
Construct the look vector from the two look angles.

look_vector[0] = tan(psi)
look_vector[1] = tan(delta)
look_vector[2] = 1.0

Convert the orbital look vector to the ACS coordinate system:

Use the roll-pitch-yaw values for this GCP to construct the orbital to ACS rotation matrix $\mathbf{M}_{ORB2ACS} = [\mathbf{M}_{ACS2ORB}]^T$.

Where: $\mathbf{M}_{ACS2ORB}$ =

$$\begin{bmatrix} \cos(p)\cos(y) & \sin(r)\sin(p)\cos(y)+\cos(r)\sin(y) & \sin(r)\sin(y)-\cos(r)\sin(p)\cos(y) \\ -\cos(p)\sin(y) & \cos(r)\cos(y)-\sin(r)\sin(p)\sin(y) & \cos(r)\sin(p)\sin(y)+\sin(r)\cos(y) \\ \sin(p) & -\sin(r)\cos(p) & \cos(r)\cos(p) \end{bmatrix}$$

Convert look_vector to ACS_look_vector by multiplying it by $[\mathbf{M}_{ACS2ORB}]^T$:

ACS_look_vector = $[\mathbf{M}_{ACS2ORB}]^T$ look_vector

Use the attitude corrections to construct the ACS correction rotation matrix $\mathbf{M}_{Precision}$:

1. Compute the precision correction at the time (t_att = att_seconds + att_time) corresponding to the attitude sample:
   a. roll_corr = roll_bias + roll_rate * (t_att – t_ref – image_seconds)
   b. pitch_corr = pitch_bias + pitch_rate * (t_att – t_ref – image_seconds)
   c. yaw_corr = yaw_bias + yaw_rate * (t_att – t_ref – image_seconds)

   Note that only the seconds of day fields are needed for the attitude and image epochs, as they are constrained to be based on the same year and day.

2. Compute the rotation matrix corresponding to roll_corr, pitch_corr, and yaw_corr ($\mathbf{M}_{Precision}$), using the same equations used for $\mathbf{M}_{ACS2ORB}$ above.

Apply the attitude corrections to the look vector by multiplying by $\mathbf{M}_{Precision}$:

ACS_pert_look_vector = $\mathbf{M}_{Precision}$ ACS_look_vector

Rotate the line-of-sight back to the orbital coordinate system, using the transpose of the $\mathbf{M}_{ORB2ACS}$ matrix, which is the same as $\mathbf{M}_{ACS2ORB}$:

pert_look_vector = $\mathbf{M}_{ACS2ORB}$ ACS_pert_look_vector

Note that this can be achieved with a single rotation of:

$\mathbf{M}_{corr} = \mathbf{M}_{ACS2ORB}\ \mathbf{M}_{Precision}\ [\mathbf{M}_{ACS2ORB}]^T$

pert_look_vector = $\mathbf{M}_{corr}$ look_vector

Calculate the new look angles:

psi = arctan(pert_look_vector[0]/pert_look_vector[2])

delta = arctan(pert_look_vector[1]/pert_look_vector[2])

Calculate Observation Residual

Correct the final values of alpha and beta for all GCPs for the effects of the final solution iteration. These values are updated by process_one_gcp for all but the final iteration.

For each GCP

Calculate the full partial coefficients matrix for alpha and beta.

For all 6 elements

Calculate the residual for alpha by subtracting the calculated observation.

gcps.va = gcps.va - H1 *Ya

Calculate the residual for beta by subtracting the calculated observation.

gcps.vb = gcps.vb - H2 *Ya

Where Ya is the vector containing the incremental parameter corrections for the last iteration.

Finish Processing

Update the OLI model file and write to the solution and residual files. This includes functions that check the solution quality statistics (pre-fit RMS, post-fit RMS, outlier percent, number of valid GCPs) to determine if the solution was successful.

Compute the percentage of GCPs that were declared outliers:

percent_outlier = num_outlier / num_GCP * 100

Compute the number of valid GCPs:

num_valid = num_GCP – num_outlier

Check the pre-fit RMS, post-fit RMS, percent_outlier, and num_valid metrics against the thresholds (maximum pre-fit RMS, maximum post-fit RMS, maximum outlier percentage, minimum number of valid GCPs) from the CPF.

If the pre- and post-fit RMS values are both below the thresholds, and either the percent_outlier metric is below threshold or the num_valid metric is above the threshold:

Update the model to make a precision model.
Fill the gcp_solution structure with the appropriate values.
Write to the solution file.
Return success status.

Else return failure status.

Update LOS Model

Update the LOS model file with the precision-correction values. The LOS model will be read from the LOS model file, the new precision correction values will be placed in the LOS model structure, the LOS model will be processed with the new precision-correction values, and the new precision LOS model structure will be output to the precision LOS model file.

Unlike the heritage approach, not only are the precision-correction parameters stored in the LOS model, they are also applied to both the ephemeris and attitude data sequences. This is captured in the LOS model by storing both original and corrected attitude and ephemeris data sequence. This update procedure operates as follows:

Correct Attitude Sub-Algorithm

Apply the ACS/body space attitude corrections computed by the LOS/precision correction procedure to the attitude data sequence. Output a parallel table of roll-pitch-yaw values with the precision corrections applied. This "corrected" table is created by the LOS Model Creation algorithm, but initially it is identical to the original attitude data sequence.

The sequence of transformations required to convert a line-of-sight in the OLI instrument coordinate system, generated using the Legendre polynomials, is as follows:

$$\underline{\mathbf{x}}_{ECEF} = \mathbf{M}_{ORB2ECEF}\,\mathbf{M}_{ACS2ORB}\,\mathbf{M}_{Precision}\,\mathbf{M}_{OLI2ACS}\,\underline{\mathbf{x}}_{OLI}$$

where:
- $\underline{\mathbf{x}}_{OLI}$ is the Legendre-derived instrument LOS vector
- $\mathbf{M}_{OLI2ACS}$ is the OLI to ACS alignment matrix from the CPF
- $\mathbf{M}_{Precision}$ is the correction to the attitude data, computed by the LOS/precision correction procedure
- $\mathbf{M}_{ACS2ORB}$ is the spacecraft attitude (roll-pitch-yaw)
- $\mathbf{M}_{ORB2ECEF}$ is the orbital-to-ECEF transformation computed using the ECEF ephemeris
- $\underline{\mathbf{x}}_{ECEF}$ is the LOS vector in ECEF coordinates

Note that in the heritage implementation, the sequence was:

$$\underline{\mathbf{x}}_{ECEF} = \mathbf{M}_{ORB2ECEF}\,\mathbf{M}_{Precision}\,\mathbf{M}_{ACS2ORB}\,\mathbf{M}_{OLI2ACS}\,\underline{\mathbf{x}}_{OLI}$$

For nadir-viewing imagery, the $\mathbf{M}_{ACS2ORB}$ matrix is nearly identical, so there is little difference. Since OLI will occasionally be viewing off-nadir and it is more natural to model attitude errors in the ACS/body coordinate system, the order has been reversed for Landsat 8/9. The impact is minimal in the model and LOS projection, but becomes more important for the LOS/precision correction algorithm.

This new sub-algorithm pre-computes the $\mathbf{M}_{ACS2ORB}$ $\mathbf{M}_{Precision}$ combination and stores the corresponding corrected roll-pitch-yaw attitude sequence in the model structure. This approach has the following advantages:

1. It streamlines the application of the model for LOS projection by removing the step of explicitly applying the precision correction.
2. It allows for the use of a more complex correction model in the future since the application of the model is limited to this unit. Note that the Earth-view attitude correction model consists of the following model parameters:
   Precision reference time: t_ref in seconds from the image epoch (nominally near the center of the image time window)
   Roll bias and rate corrections: roll_bias, roll_rate
   Pitch bias and rate corrections: pitch_bias, pitch_rate
   Yaw bias and rate corrections: yaw_bias, yaw_rate
   This model is dealt with in more detail in the line-of-sight correction algorithm description.
3. Retaining both the original and corrected attitude sequences in the model makes the model self-contained and will make it unnecessary for the LOS/precision correction algorithm to access the preprocessed ancillary data.

The disadvantage is that it doubles the size of the attitude data in the model structure.

The construction of the corrected attitude sequence proceeds as follows:

For each point in the attitude sequence j = 0 to K-1:

1. Compute the rotation matrix corresponding to the j^{th} roll-pitch-yaw values:

$\mathbf{M}_{ACS2ORB}$ =

$$\begin{bmatrix} \cos(p)\cos(y) & \sin(r)\sin(p)\cos(y)+\cos(r)\sin(y) & \sin(r)\sin(y)-\cos(r)\sin(p)\cos(y) \\ -\cos(p)\sin(y) & \cos(r)\cos(y)-\sin(r)\sin(p)\sin(y) & \cos(r)\sin(p)\sin(y)+\sin(r)\cos(y) \\ \sin(p) & -\sin(r)\cos(p) & \cos(r)\cos(p) \end{bmatrix}$$

2. Compute the precision correction at the time (t_att = att_seconds + att_time(j)) corresponding to the attitude sample:
   a. roll_corr = roll_bias + roll_rate * (t_att – t_ref – image_seconds)
   b. pitch_corr = pitch_bias + pitch_rate * (t_att – t_ref – image_seconds)
   c. yaw_corr = yaw_bias + yaw_rate * (t_att – t_ref – image_seconds)

   Note that only the seconds of day fields are needed for the attitude and image epochs, as they are constrained to be based on the same year and day.
3. Compute the rotation matrix corresponding to roll_corr, pitch_corr, and yaw_corr ($\mathbf{M}_{Precision}$), using the same equations presented in step 1 above.

4. Compute the composite rotation matrix: $\mathbf{M} = \mathbf{M}_{ACS2ORB}\ \mathbf{M}_{Precision}$
5. Compute the composite roll-pitch-yaw values:

$$roll' = -\tan^{-1}\left(\frac{M_{2,1}}{M_{2,2}}\right)$$

$$pitch' = \sin^{-1}\left(M_{2,0}\right)$$

$$yaw' = -\tan^{-1}\left(\frac{M_{1,0}}{M_{0,0}}\right)$$

6. Store the composite roll'-pitch'-yaw' values in the j^{th} row of the corrected attitude data table.

Correct Ephemeris Sub-Algorithm

The heritage algorithm converted the ephemeris information (position and velocity) from the Earth-Centered Inertial (ECI J2000) system to the ECEF system and applied the ephemeris corrections computed in the LOS/precision correction procedure to both ephemeris sets. Since both ECI and ECEF representations of the ephemeris are now provided by the Ancillary Data Preprocessing Algorithm (Section 4.1.4), the first portion of the heritage algorithm is no longer necessary.

The precision-correction parameters are stored in the LOS model in the spacecraft orbital coordinate system as three position (x_bias, y_bias, z_bias) corrections and three velocity (x_rate, y_rate, z_rate) corrections that, like the attitude corrections, are relative to t_ref. These values must be converted to the ECEF and ECI coordinate systems. Once the precision correction is determined in the ECEF/ECI coordinate system, the ECEF/ECI ephemeris values can be updated with the precision parameters.

Loop on LOS model ephemeris points j = 0 to N-1

Compute the precision correction:

Calculate delta time for precision correction:

dtime = ephem_seconds + ephem_time(j) – t_ref – image_seconds

Calculate the change in X, Y, Z due to precision correction. Corrections are in terms of spacecraft orbital coordinates.

dx orb = model precision x_bias + model precision x_rate * dtime
dy orb = model precision y_bias + model precision y_rate * dtime
dz orb = model precision z_bias + model precision z_rate * dtime

where:
model precision x_bias = precision (orbital coord sys) update to X position
model precision y_bias = precision (orbital coord sys) update to Y position
model precision z_bias = precision (orbital coord sys) update to Z position

model precision x_rate = precision (orbital coord sys) update to X velocity
model precision y_rate = precision (orbital coord sys) update to Y velocity
model precision z_rate = precision (orbital coord sys) update to Z velocity

Construct precision position and velocity “delta” vectors.

$$[\text{dorb}] = \begin{bmatrix} \text{dx orb} \\ \text{dy orb} \\ \text{dz orb} \end{bmatrix}$$

$$[\text{dvorb}] = \begin{bmatrix} \text{model precision x rate} \\ \text{model precision y rate} \\ \text{model precision z rate} \end{bmatrix}$$

Calculate the orbit to ECF transformation [ORB2ECEF] using ECEF ephemeris (See the Ancillary Data Preprocessing Algorithm Section 4.1.4 for this procedure).

Transform precision “delta” vectors to ECEF.

$$[\text{def}] = [\text{ORB2ECEF}][\text{dorb}]$$

$$[\text{dvef}] = [\text{ORB2ECEF}][\text{dvorb}]$$

Adjust ECEF ephemeris by the appropriate “delta” precision vector and store the new ephemeris in the model. These ephemeris points will be used when transforming an input line/sample to an output projection line/sample.

Model EF Position = Ephemeris EF Position + def
Model EF Velocity = Ephemeris EF Velocity + dvef

where:

All parameters are 3x1 vectors
ephemeris ecef values are the interpolated one-second ephemeris values in ECEF coordinates

Calculate the orbit to ECI transformation [ORB2ECI] using ECI ephemeris.

Transform precision “delta” vectors to ECI.

$$[\text{deci}] = [\text{ORB2ECI}][\text{dorb}]$$

$$[\text{dveci}] = [\text{ORB2ECI}][\text{dvorb}]$$

Adjust ECI ephemeris by the appropriate “delta” precision vector and store the new ephemeris in the model. These ephemeris points will be used with lunar/stellar observations.

Model ECI Position = Ephemeris ECI Position + deci
Model ECI Velocity = Ephemeris ECI Velocity + dveci

where:

All parameters are 3x1 vectors
ephemeris eci values are the interpolated one-second ECI ephemeris

Convert the Net Attitude Corrections to Alignment Angles

This sub-algorithm combines the newly computed attitude correction with the OLI sensor alignment matrix from the LOS model to construct corrected alignment angles.

Compute the precision correction at the reference time t_ref:

roll_corr = roll_bias
pitch_corr = pitch_bias
yaw_corr = yaw_bias

Compute the rotation matrix corresponding to roll_corr, pitch_corr, and yaw_corr ($\mathbf{M}_{Precision}$), using the standard rotation matrix equations:

$\mathbf{M}_{Precision}$ =

$$\begin{bmatrix} \cos(p)\cos(y) & \sin(r)\sin(p)\cos(y)+\cos(r)\sin(y) & \sin(r)\sin(y)-\cos(r)\sin(p)\cos(y) \\ -\cos(p)\sin(y) & \cos(r)\cos(y)-\sin(r)\sin(p)\sin(y) & \cos(r)\sin(p)\sin(y)+\sin(r)\cos(y) \\ \sin(p) & -\sin(r)\cos(p) & \cos(r)\cos(p) \end{bmatrix}$$

Extract the ACS to OLI alignment matrix, $\mathbf{M}_{ACS2OLI}$, from the OLI LOS model, and take the transpose to compute $\mathbf{M}_{OLI2ACS}$.

Compute the composite alignment matrix: $\mathbf{M} = \mathbf{M}_{Precision}\,\mathbf{M}_{OLI2ACS}$

Compute the composite roll-pitch-yaw alignment angles:

$$\text{roll}' = -\tan^{-1}\left(\frac{M_{2,1}}{M_{2,2}}\right)$$

$$\text{pitch}' = \sin^{-1}\left(M_{2,0}\right)$$

$$\text{yaw}' = -\tan^{-1}\left(\frac{M_{1,0}}{M_{0,0}}\right)$$

Extract the orbital ephemeris biases from the precision solution: (x_bias, y_bias, z_bias).

Extract the attitude bias correction and ephemeris bias correction covariance terms from the precision solution covariance matrix:

The solution provides:

$$X = \begin{bmatrix} droll \\ dpitch \\ dyaw \\ dX \\ dY \\ dZ \\ droll\,/\,dt \\ dpitch\,/\,dt \\ dyaw\,/\,dt \\ dX\,/\,dt \\ dY\,/\,dt \\ dZ\,/\,dt \end{bmatrix}$$ with covariance Cov

We want to form:

$$X = \begin{bmatrix} droll \\ dpitch \\ dyaw \\ dX \\ dY \\ dZ \end{bmatrix}$$

With (see note #2):

$$CovX = \begin{bmatrix} Cov[0][0] & Cov[0][1] & Cov[0][2] & Cov[0][3] & Cov[0][4] & Cov[0][5] \\ Cov[1][0] & Cov[1][1] & Cov[1][2] & Cov[1][3] & Cov[1][4] & Cov[1][5] \\ Cov[2][0] & Cov[2][1] & Cov[2][2] & Cov[2][3] & Cov[2][4] & Cov[2][5] \\ Cov[3][0] & Cov[3][1] & Cov[3][2] & Cov[3][3] & Cov[3][4] & Cov[3][5] \\ Cov[4][0] & Cov[4][1] & Cov[4][2] & Cov[4][3] & Cov[4][4] & Cov[4][5] \\ Cov[5][0] & Cov[5][1] & Cov[5][2] & Cov[5][3] & Cov[5][4] & Cov[5][5] \end{bmatrix}$$

The covariance matrix captures the correlations between the attitude and ephemeris correction parameters (e.g., roll-Y and pitch-X).

The following fields are output to the alignment characterization database:

Reference time: image epoch year, image epoch day, image epoch second + t_ref

Alignment vector: X above

Alignment covariance: CovX above
RMS GCP fit
Number of GCPs used
Outlier threshold used
Scene off-nadir roll angle
Control type flag (DOQ or GLS)
L0Rp ID
Work Order ID
WRS Path/Row

LOS Model Correction Output Summary

The primary output of the LOS model correction algorithm is the updated "precision" LOS model. This model has the same structure as the input LOS model, which is described in the LOS Model Creation Algorithm (Section 4.2.1). Although the model structure is the same, the corrected ECI position and velocity and the corrected ECEF position and velocity sections of the Ephemeris Model, the corrected roll-pitch-yaw section of the Attitude Model, and the Precision Correction Model all contain updated values as a result of the LOS model correction algorithm.

Table *4-10.* LOS Model Correction Solution Output File ContentsTable 4-10 shows the contents of the output LOS Model Correction Solution File. This report file documents the results of the LOS model correction procedure. It contains standard header fields common to all geometric report files.

Field	Description
Date and time	Date (day of week, month, day of month, year) and time of file creation.
Spacecraft and instrument source	Landsat 8/9 and OLI
Processing Center	EROS Data Center SVT
Work order ID	Work order ID associated with processing (blank if not applicable)
WRS path/row	WRS path and row
Software version	Software version used to create report
Off-nadir angle	Off-nadir roll angle of processed image file
Acquisition Type	Earth viewing or Lunar
L0Rp ID	Input L0Rp image ID
L1G image file	Name of L1G used to measure GCPs
Precision solution reference time	Time reference for model correction parameters as year, day of year and seconds of day.
Roll-pitch-yaw attitude corrections	Attitude bias corrections in microradians
Roll-pitch-yaw rate corrections	Attitude rate corrections in microradians/second
Roll-pitch-yaw standard deviations	Attitude bias parameter sigmas in microradians
Roll-pitch-yaw rate std. devs.	Attitude rate parameter sigmas in microrads/sec
Ephemeris position corrections	Ephemeris X-Y-Z bias corrections in meters
Ephemeris velocity corrections	Ephemeris Vx-Vy-Vz corrections in meters/second
Position standard deviations	Ephemeris X-Y-Z sigmas in meters
Velocity standard deviations	Ephemeris Vx-Vy-Vz sigmas in meters/second
Across-track covariance matrix	6-by-6 covariance matrix for roll, Y, Z, roll rate, Vy, Vz correction parameters.

Along-track covariance matrix	6-by-6 covariance matrix for X, pitch, yaw, Vx, pitch rate, yaw rate correction parameters.
Spacecraft roll-pitch-yaw at solution reference time	Spacecraft attitude at solution reference time in microradians

Table 4-10. LOS Model Correction Solution Output File Contents

Table 4-11 shows the contents of the LOS Model Correction Residuals file. This file documents the GCP residuals for the final set of GCPs (after the outlier rejection loop has found no additional outliers), including the residuals for each iteration of the weighted least-squares solution procedure. Thus, it contains both the initial (pre-correction) and final (post-correction) residuals. This file is used as an input by the Geodetic Accuracy Assessment algorithm. The output residual file also contains the standard report header mentioned above.

Field	Description
Date and time	Date (day of week, month, day of month, year) and time of file creation.
Spacecraft and instrument source	Landsat 8/9 and OLI
Processing Center	EROS Data Center SVT
Work order ID	Work order ID associated with processing (blank if not applicable)
WRS path/row	WRS path and row
Software version	Software version used to create report
Off-nadir angle	Off-nadir roll angle of processed image file
Acquisition Type	Earth viewing or Lunar
L0Rp ID	Input L0Rp image ID
L1G image file	Name of L1G used to measure GCPs
Number of GCPs used	Number of valid (non-outlier) GCPs
Heading for individual GCPs	One line of ASCII text containing column headings for the individual GCP fields.
For each iteration:	
Iteration number	Starts with 0 for initial (uncorrected) residuals and ends with "Final" for results of last iteration.
For each GCP:	
Point ID	GCP ID (see GCP Correlation Algorithm for details)
Predicted L1G Line	Predicted L1G line location
Predicted L1G Sample	Predicted L1G sample location
GCP Time of Observation	Seconds from image epoch time
Latitude	GCP latitude in degrees
Longitude	GCP longitude in degrees
Height	GCP height in meters
Across-track Angle (delta)	Across-track LOS angle in degrees
Across-track Residual	Residual on delta converted to meters
Along-track Residual	Residual on psi converted to meters
Y Residual	Residual in Y/line direction in meters
X Residual	Residual in X/sample direction in meters
Outlier Flag	0 for outlier, 1 for valid GCP
GCP Source	DOQ or GLS

Table 4-11. LOS Model Correction Residuals Output File Contents

Table 4-12 lists the fields stored in the characterization database for future sensor alignment calibration operations.

Field	Description
Work order ID	Work order ID associated with processing
WRS path/row	WRS path and row
L0Rp ID	Input L0Rp image ID
Control Type	DOQ or GLS
Off-nadir angle	Off-nadir roll angle of scene (in degrees)
Number of GCPs used	Number of valid (non-outlier) GCPs
Outlier threshold used	Confidence level used for outlier rejection threshold
RMS GCP Fit	RMS of final iteration across- and along-track residuals in meters. This field would subsequently be used to identify entries that may be suspect due to poor fits to the ground control.
Precision solution reference time	Time reference for model correction parameters as year, day of year and seconds of day.
Roll-pitch-yaw alignment angles	Composite alignment angles in microradians
Ephemeris position corrections	Ephemeris X-Y-Z bias corrections in meters
Alignment covariance matrix	6-by-6 covariance matrix for roll, pitch, yaw, X, Y, Z correction parameters.

Table 4-12. Model/Alignment Characterization Database Output Fields

Table 4-13 lists the fields stored in the characterization database to support future GCP quality assessment and improvement activities (see note #3). Only the residuals for non-outlier GCPs from the initial (zeroth) iteration are written to the characterization database.

Field	Description
For each GCP:	
Work order ID	Work order ID associated with processing
WRS path/row	WRS path and row
L0Rp ID	Input L0Rp image ID
Point ID	GCP ID (see the GCP Correlation Algorithm for details)
GCP Time of Observation	Year, day of year, and seconds of day
Ephemeris Position	Spacecraft ECEF position at GCP time (meters)
Ephemeris Velocity	Spacecraft ECEF velocity at GCP time (meters/sec)
Spacecraft Roll-Pitch-Yaw	Spacecraft roll-pitch-yaw at GCP time (radians)
True Latitude	GCP latitude in radians
True Longitude	GCP longitude in radians
True Height	GCP height in meters
Apparent Latitude	Latitude measured in L1G image in radians
Apparent Longitude	Longitude measured in L1G image in radians
Apparent Height	Height interpolated from DEM in meters
GCP Source	DOQ or GLS

Table 4-13. GCP Residual Characterization Database Output Fields

4.2.3.8 Notes

Some additional background assumptions and notes include the following:

1. The heritage aliprecision process uses the DDR for the L1G mensuration image to retrieve the image framing and projection parameter information necessary to convert output space line/sample coordinates to latitude/longitude, but the same information is available in the grid file, so either could be used.
2. The extent to which the model creation logic must be rerun was scaled back as compared to the heritage implementation. The precision ephemeris corrections are embedded in the model ephemeris so it must be regenerated, but full model reprocessing is not truly necessary. This was done in the past for convenience (because it is fast and was easy to simply invoke the model creation logic as a subroutine – Update LOS Model).
3. Possible additional solution quality metrics include initial vs. final GCP distribution metrics, but are not implemented in the baseline version.
4. The heritage ALIAS (and Landsat) LOS model correction algorithms required the L1R file as input data to provide the L1R input space image dimensions. This information will now be available in the LOS model in the image sub-model, so the L1R input is no longer required.
5. The logic that computes the OLI sensor alignment corrections implied by the precision attitude and ephemeris corrections was added to this algorithm to ensure that the computed corrections are applied to the proper sensor alignment matrix. Storing only the corrections leaves open the question of what alignment they are relative to. This was not a problem in the heritage systems (L7 IAS, ALIAS) because the alignment calibration process was run as one continuous flow, using the same set of data throughout. This approach limited the number of scenes that could be processed and restricted the order of processing to be in data acquisition order. This restriction will be lifted for OLIAS so a much larger volume of data can be reduced and trended for subsequent offline analysis. This requires the trended data to be converted to apparent alignment angles so acquisitions processed using different alignment calibrations can be compared.
6. The covariance data that are trended for subsequent use in alignment calibration are a subset of the full precision solution covariance.
7. Trending of a slightly enhanced version of the initial (zeroth) iteration GCP residuals has been added to support offline research into large-scale area triangulation. The path/row, date/time, GCP ID, true position, and apparent (mensuration image) position are recorded for all non-outlier GCPs.
8. Since the precision correction process will likely be run prior to any cloud screening and will therefore frequently fail due to the inability to correlate GCPs in cloud-covered imagery, thresholds and bounds will need to be developed to detect cases in which the solution has failed. In this case, scene processing would fail over to a terrain-corrected systematic data flow. The prototype computes and reports three quality metrics, including prefit GCP RMS, postfit GCP RMS, and percent GCP outliers, but does not apply any threshold logic. The operational version should apply thresholds on the pre-fit and post-fit GCP RMSE values, and make sure that either a sufficient number of valid GCPs were used or that the percentage of GCPs declared outliers was not too high.
9. Using a systematic terrain-corrected image for GCP mensuration instead of the heritage systematic image required some modifications to the GCP processing

logic. Specifically, the DEM elevation associated with the measured GCP image position is used to construct the “apparent” LOS instead of using zero for a LOS projected to the ellipsoid surface as the heritage algorithm does. Note that, while the actual GCP elevation could be used, this would introduce error that would grow with the misregistration between the systematic image and the DEM, making the simpler approach less robust. This change was motivated by the large GCP search areas that would be required in systematically corrected images for off-nadir scenes in high elevation areas.

10. Using the DEM as the source of “apparent” GCP height allows the algorithm to support either terrain-corrected or systematic image inputs. If an input DEM is not provided, the “apparent” GCP height will be set to zero as it is now. If an input DEM is provided, it will be used as the source of the “apparent” GCP height. Note that the capability to use SCA-combined mensuration images only applies to terrain-corrected images.

4.2.4 OLI Resampling Algorithm

4.2.4.1 Background/Introduction

The OLI resampling method is used to take an L1R image, which has unevenly spaced pixels with respect to the surface of the object imaged, and create a reprojected image where all image pixels are located within an evenly spaced set of grid points, or output space, with respect to the object imaged. This mapping is subject to the errors associated with the interpolation method used to determine the digital numbers associated with the output image.

The geometric resampling grid and the geometric model are used to calculate the mappings between the input and output space. The geometric model contains the individual detector offsets from a nominal location, while the geometric resampling grid contains all other mapping variables. The resampling grid provides a mapping from a 2D input space to a 3D output space, and vice versa. The output space corresponds to x/y/z projection locations, while the input space corresponds to line/sample locations within the L1R. The z component in output space is elevation. If elevation is not to be accounted for during processing, an elevation of zero is used to map output pixels to input pixels.

Due to what can be rather large sample-to-sample offsets within an L1R image, the cubic convolution interpolation option works in a two-step process. A hybrid set of pixels in the sample direction are created using cubic convolution resampling in the line direction. This creates a set of unevenly spaced pixels in the sample direction. The Akima A interpolation method is then used to determine the final digital number for the output image by resampling the hybrid pixels in the sample direction. The nearest-neighbor resampling option simply determines the closest input pixel for the corresponding output pixel.

The USGS defines products L1G, radiometrically calibrated with only systematic geometric corrections, L1GT, radiometrically calibrated systematic corrections and the use of a Digital Elevation Model (DEM) to correct for relief displacement, and L1TP,

radiometrically calibrated and orthorectified using ground control points and a DEM to correct for relief displacement. When necessary, this algorithm will specifically designate the types of products to which the section applies. If the reference is to any of the three product types available then the designation L1G will be used.

4.2.4.2 Dependencies

The OLI resampling algorithm assumes that the Ancillary Data Preprocessing, LOS Model Creation, and Line-of-Sight Projection to Ellipsoid and Terrain algorithms have been executed, and an L1R has been generated. If a DEM is given as input to account for relief, or terrain, displacement the grid must have an adequate number and range (elevation bounds) of z-planes to cover the entire elevation range within the L1R. A geometric model and grid must be available for the L1R. The Ancillary Data Preprocessing (4.1.4), LOS Model Creation (4.2.1), and Line-of-Sight Projection/Grid Generation (4.2.2) Algorithms contain more information about the data structure and contents of the Geometric Model and Resampling Grid.

4.2.4.3 Inputs

The resampling algorithm and its component sub-algorithms use the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs
L1R Image
Resampling Grid (see the Line of Sight Projection Algorithm for contents)
Bands to process
Terrain correction Flag (yes/no)
DEM (if terrain flag set to yes)
SCA combine flag (yes/no)
Geometric model (see Line of Sight Model Creation Algorithm for contents)
Resampling type (CC,NN)
Minimum and maximum DN of output
Output data type
α (if resampling type is CC) (defaults to -0.5)
Fill pixel value

4.2.4.4 Outputs

Resampled output image (L1GS, L1GT, or L1TP)
Resampled image data (band separated, either SCA combined or SCA separated)
Image data metadata fields (see Table 4-14)

4.2.4.5 Options

Cubic convolution or nearest-neighbor resampling
Creating an output image with Sensor Chip Assemblies (SCAs) combined or separated
Applying terrain correction, yes or no

4.2.4.6 Procedure

OLI resampling interpolates radiometrically corrected but geometrically raw image data to a map-projected output space. The resampling process uses information stored in the

resampling grid, along with focal plane calibration data stored in the geometric model, to map output projection locations to an input location. Since an input location for an output pixel typically lies at a non-integer location, interpolation is used to find the pixel value associated with this non-integer location. OLI resampling is performed on the geometrically raw L1R data using one of two methods: cubic convolution combined with the Akima A method, or nearest neighbor. Note that Modulation Transfer Function Compensation (MTFC) and bilinear resampling are not supported in the baseline algorithm. Due to the lack of inherent band registration and the need to perform subpixel registration to achieve OLI band alignment, cubic convolution, combined with the Akima A interpolation method, will be used to generate the standard L8/9 products. It is also important to have the best subpixel accuracy in the output image during geometric characterization and calibration, so cubic convolution is chosen for interpolation during the characterization and calibration of the OLI instrument. The ALIAS-heritage nearest-neighbor interpolation capability is also provided as an option for special-purpose science products and testing purposes. Since this document focuses on both standard product generation and geometric characterization and calibration, the only interpolation method discussed in detail here is the cubic convolution combined with the Akima A method.

During resampling, there is a need to know what input pixel location corresponds to a given output pixel location. The OLI geometric processing system does not have a “true” inverse model to perform this calculation. Instead, for a given output pixel, the corresponding input pixel is found from the forward and inverse mapping coefficients stored in the resampling grid. There are two scenarios when performing this calculation. The first involves performing resampling for a systematic image, in which case the dimension for z, or elevation, is either zero or a constant value. This involves only a two-dimensional operation in line and sample. The second involves performing resampling for a terrain-corrected image. A terrain-corrected image has the effects of relief removed from the output imagery. When working with a terrain-corrected image, a three-dimensional operation is performed during the inverse mapping, with the dimensions being input (L1R) line, input sample, and elevation (Figure 4-33). Both procedures of mapping output pixel locations to input pixel locations are discussed below.

Due to the layout of the OLI focal plane, there are along-track offsets between spectral bands within each SCA, along-track offsets between even and odd SCAs, and a reversal of the band ordering in adjacent SCAs. This leads to an along-track offset in the imagery coverage area for a given band between odd and even SCAs, as well as an offset between bands within each SCA. To create a more uniform image coverage within a geometrically corrected output product, the leading and trailing imagery associated with these offsets is trimmed. This trimming is controlled by a set of latitude/longitude bounds for the active image area for each band, contained in the input resampling grid. Trimming is implemented by converting these bounds to a look up table that lists the starting and ending sample location of active (non-fill) data for each line of the output image.

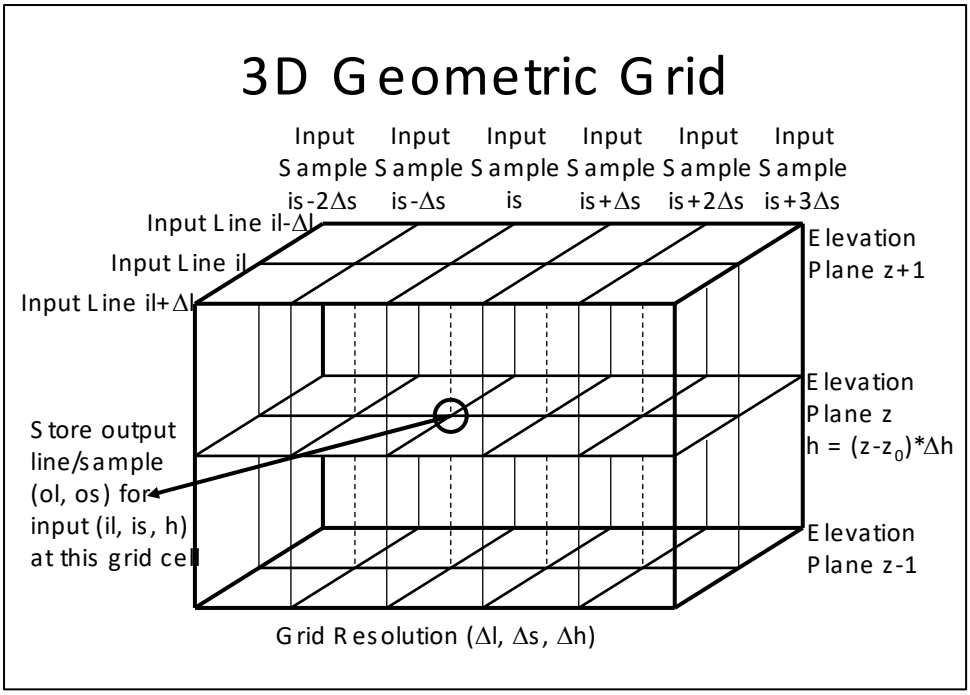

Figure 4-33. 3D Grid Representation

4.2.4.6.1 Using the geometric grid to map an output pixel location to an input pixel location.

To find an input line/sample location for an output line/sample location, given that the elevation is zero:

1. Calculate an input line and sample location using the rough polynomial stored in the resampling grid and the current output line and sample location.

$$\text{approximate input line} = \sum_{n=0}^{N}\left[\sum_{m=0}^{M}\left(ra_{n,m} * (\text{output sample})^{m}\right) * (\text{output line})^{n}\right]$$

$$\text{approximate input sample} = \sum_{n=0}^{N}\left[\sum_{m=0}^{M}\left(rb_{n,m} * (\text{output sample})^{m}\right) * (\text{output line})^{n}\right]$$

Where:

ra = rough polynomial mapping coefficients for line mapping
rb = rough polynomial mapping coefficients for sample mapping
M = Number of sample coefficients in the polynomial
N = Number of line coefficients in the polynomial

Previous experience when working with the ALI instrument, has demonstrated a 1st order polynomial in both the line and sample direction will suffice for the rough polynomial, thus M = N = 1.

$$\text{approximate input line} = a_0 + a_1 * \text{output sample} + a_2 * \text{output line} + a_3 * \text{output sample} * \text{output line}$$
$$\text{approximate input sample} = b_0 + b_1 * \text{output sample} + a_2 * \text{output line} + a_3 * \text{output sample} * \text{output line}$$

2. Calculate the grid cell location for the approximate input line and sample location.

$$\text{row} = \frac{\text{approximate input line}}{\text{number of lines per cell}}$$

$$\text{column} = \frac{\text{approximate input sample}}{\text{number of samples per cell}}$$

Where:
number of lines per cell = size of the grid cell in lines
number of samples per cell = size of the grid cell in samples

Set this grid cell column and row location as the current grid cell column and row location.

3. Using the current grid cell location, check if the correct grid cell has been found.

Use input (current) mapping grid cell coefficients (a_i and b_i) to map output line and sample to input:

input line = $b_0 + b_1 *$ output sample + $b_2 *$ output line + $b_3 *$ output line * output sample
input sample = $a_0 + a_1 *$ output sample + $a_2 *$ output line + $a_3 *$ output line * output sample

Calculate the grid cell location for this input line and sample location:

$$\text{new row} = \frac{\text{input line}}{\text{number of lines per cell}}$$
$$\text{new column} = \frac{\text{input sample}}{\text{number of samples per cell}}$$

If the new grid cell (new row and new column) is the same as the current grid cell (current row and current column):

The correct grid cell has been found, inverse grid mapping coefficients for this grid cell are used to calculate the input line/sample for the current output line/sample.

If the new grid cell (new row and new column) is not the same as the current grid cell (current row and current column):

The new grid cell is chosen as current grid cell, and the 3rd step is repeated until the correct grid cell is found.

This routine or function listed above, of mapping output pixel locations to input pixel locations without taking into account elevation, will be referred to as ols2ils (output space line-sample to input space line-sample mapping). The ols2ils sub-algorithm takes a given output line and sample location and calculates the grid cell column and row location, along with the corresponding input line and sample location for that output location.

To find an input line/sample location for an output line/sample location, given that the elevation is not zero:

1. Find the z planes that the elevation associated with the output pixel falls between.

$$z\ \text{plane} = (\text{int})\text{floor}\left(\frac{\text{elevation}}{\text{elevation increment}}\right) + z_{elev=0}$$

Where:
elevation = elevation associated with current output location (from DEM)
elevation increment = elevation increment between z planes stored in grid
$z_{elev=0}$ = zero z plane, the index of the zero elevation z-plane

The output line/sample falls between z plane and z plane+1.

2. Call ols2ils for z plane and z plane+1. This yields input $sample_0$, input $line_0$ and input $sample_1$, input $line_1$.
3. Interpolate between z plane and z plane + 1 to find input line and sample location for elevation.

Calculate elevations for z plane and z plane + 1:
$elev_0$ = elevation increment * (z plane - zero z plane)
$elev_1$ = $elev_0$ + elevation increment

Calculate weights for ols2ils results:

$$w_0 = \frac{elev_1 - \text{elevation}}{elev_1 - elev_0}$$

$$w_1 = \frac{\text{elevation} - elev_0}{elev_1 - elev_0}$$

input sample = input $sample_0$ * w_0 + input $sample_1$ * w_1
input line = input $line_0$ * w_0 + input $line_1$ * w_1
Where:
input $sample_0$ = input sample for z plane
input $sample_1$ = input sample for z plane + 1
input $line_0$ = input line for z plane
input $line_1$ = input line for z plane + 1

This routine or function listed above, which performs the three-dimensional output space line-sample to input space line-sample mapping, is referred to as 3d_ols2ils.

4.2.4.6.2 Resampling Methodology

The along- and cross- track detector offsets are applied during resampling. These include both the dynamic odd and even terrain-dependent relief and parallax effects that were calculated during the resampling grid generation, and the individual detector selection shift that are stored in the OLI geometric model. The nature of these geometric effects due to the individual detector characteristics is such that, in input space, they are evenly spaced in the line direction, but unevenly spaced in the sample direction. This is because as you move along raw imagery in the line direction, the detector number does not change. Since the detector number does not change along the line direction in raw input space, the along-track detector offset, stored within the geometric model, does not change. These geometric effects, due to these detector offsets, are slowly varying in time, staying essentially constant within the area that resampling is performed. Therefore, the along-track geometric effect, and essentially spacing in the line direction, can be treated as a constant over this area. The same logic helps explain why the across-track detector offset is not constant in the sample direction, since each sample comes from a different detector. This creates unevenly spaced samples in raw input space. Figure 4-34 shows an example of a detector layout and its associated offset. The squares in Figure 4-34 represent a location of an input pixel, taking into account the detector offsets. The circle with the cross hairs in Figure 4-34 represents the true input location for the current output pixel. An interpolated value is needed at this location to represent the current output pixel.

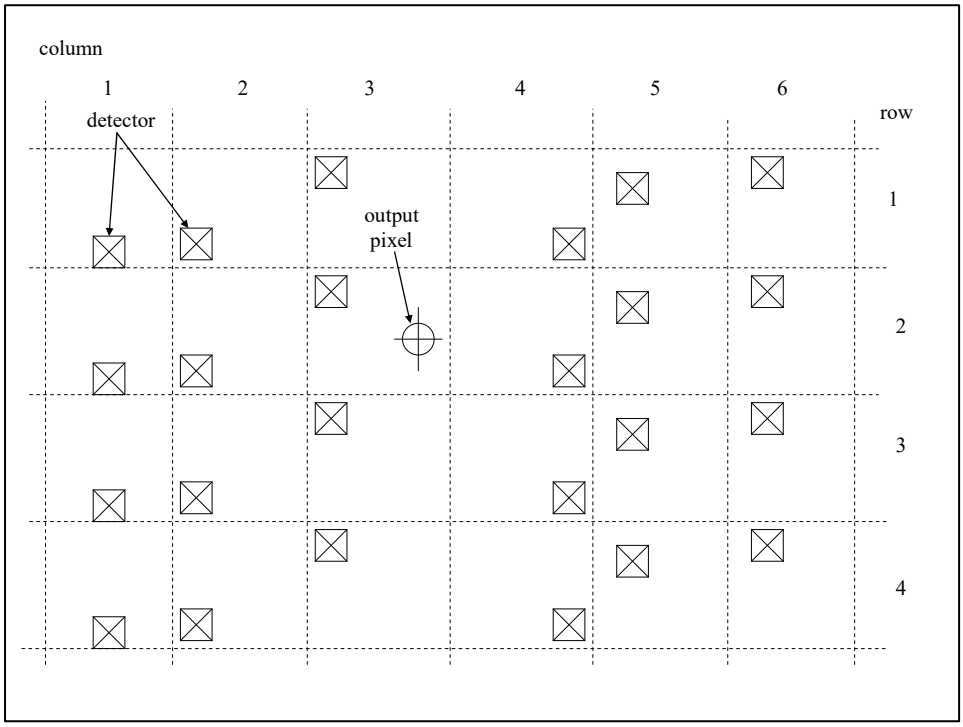

Figure 4-34. Example Detector Layout

Detector offsets are handled in the resampler by first applying a resampling kernel in the line direction that assumes evenly spaced detectors. Cubic convolution interpolation is used in the line direction; this will create a set of interpolated pixels that are aligned in the line direction. Once the pixels are aligned in the line direction, with possibly uneven spacing in the sample direction, the Akima A interpolation is used to find the final output pixel value.

Cubic convolution interpolation uses a set of piecewise cubic spline interpolating polynomials. The polynomials have the following form:

$$f(x) = \begin{cases} (\alpha+2)|x|^3 - (\alpha+3)|x|^2 + 1 & 0 \le |x| < 1 \\ \alpha|x|^3 - 5\alpha|x|^2 + 8\alpha|x| - 4\alpha & 1 \le |x| < 2 \\ 0 & |x| > 2 \end{cases}$$

Four points, centered on the point to be interpolated, are used in interpolation. The weights for each point are generated from *f(x)*. The α in the cubic convolution function is a variable parameter that affects the edge slope of the function. For standard processing, a value of -0.5 is used. Figure 4-35 shows an example of what the cubic convolution function looks like, and the corresponding weights for a phase shift of zero (marked as Xs).

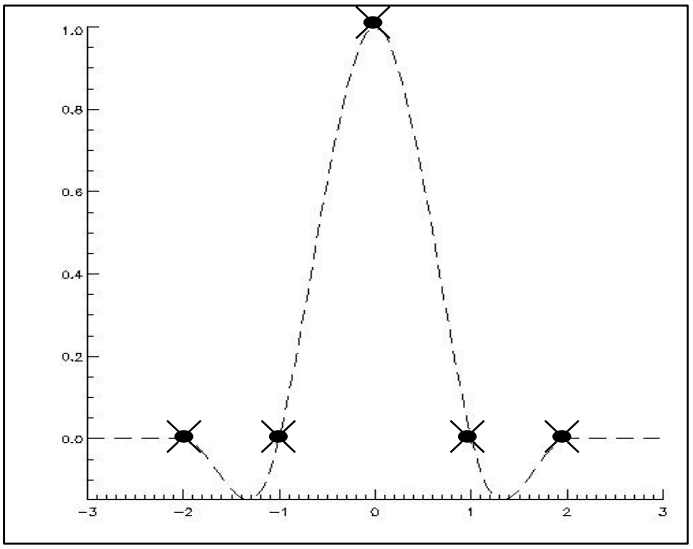

Figure 4-35. Cubic Convolution Function

As stated previously, for the OLI resampler, the cubic convolution resampling process produces a set of hybrid points that are aligned in the line direction. This is done by resampling several sets of L1R pixels in the line direction using the cubic convolution kernel; each time cubic convolution is performed, one hybrid pixel is produced. The set of hybrid points produced from the cubic convolution process is not evenly spaced in the

sample direction. Figure 4-36 illustrates a set of hybrid samples that has been aligned in the line direction using the cubic convolution process.

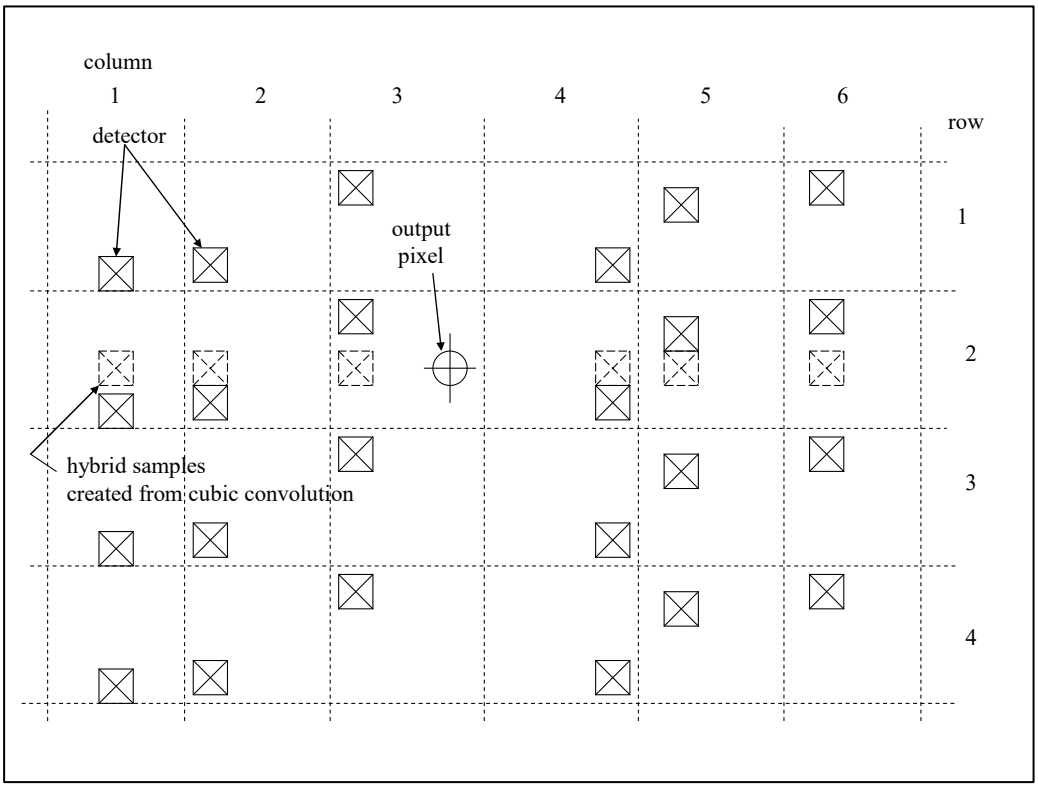

Figure 4-36. Hybrid Pixels for Detector Offsets

The Akima A method for interpolation is used for interpolating the hybrid pixels created from the cubic convolution process. This method of interpolation does not require the samples used to be evenly spaced. The Akima A method uses a third order polynomial for interpolation. The interpolating polynomial is defined by the coordinates and the slopes of the two points that are on either side of the point to be interpolated. The slopes of the adjacent points are determined as follows:

If five points are defined as 1, 2, 3, 4, and 5, then the slope at point 3, t, is defined as follows:

$$t = \frac{|m_4 - m_3| m_2 + |m_2 - m_1| m_3}{|m_4 - m_3| + |m_2 - m_1|}$$

Where:

m_1 = slope of line segment defined by points 1 and 2
m_2 = slope of line segment defined by points 2 and 3
m_3 = slope of line segment defined by points 3 and 4
m_4 = slope of line segment defined by points 4 and 5

The Akima A method of interpolation is based on the values (*y*) and slopes (*t*) on either side of the point that is to be interpolated. The interpolating polynomial for a point x between x_i and x_{i+1} is then defined as follows:

$$y = y_i + t_i * (x - x_i) + \frac{3\frac{y_{i+1} - y_i}{x_{i+1} - x_i} - 2t_i - t_{i+1}}{x_{i+1} - x_i} * (x - x_i)^2 + \frac{t_i + t_{i+1} - 2\frac{y_{i+1} - y_i}{x_{i+1} - x_i}}{(x_{i+1} - x_i)^2} * (x - x_i)^3$$

Where:

x = sample location of point to be interpolated
x_i = location of point to the left of x
x_{i+1} = location of point to the right of x
y_i = DN value for the input point at x_i
y = interpolated DN value for an output line and sample location

This methodology must be adjusted somewhat to account for higher frequency image distortion effects than those that can be captured by the conventional resampling grid. To model such effects, the L8/9 attitude data stream is separated into low-frequency and high-frequency segments, with the low-frequency portion being used for the OLI line-of-sight projection operations that build the resampling grid. The high-frequency data are interpolated to match the OLI panchromatic band line sampling times and stored in the OLI LOS model in a jitter table for application as an extra correction at image resampling time. The OLI Line-of-Sight Model Creation Algorithm (4.2.1) describes the process of separating the attitude data stream by frequency.

Sensitivity coefficients that relate these high-frequency roll-pitch-yaw jitter terms to the equivalent input image space line and sample offset effects are stored in the OLI LOS grid. This makes it possible to look up the roll-pitch-yaw jitter for each image line being resampled, and convert the jitter values to compensating input line/sample corrections that are used to refine the image interpolation location coordinates. The OLI Line-of-Sight Projection/Grid Generation Algorithm (see 4.2.2) describes the generation of these sensitivity coefficients. The process by which the jitter table from the OLI model and jitter sensitivity coefficients from the OLI grid are used during image resampling is shown schematically in Figure 4-37. The items in green in the figure are new structures, compared to the ALIAS-heritage code and processes, added to support jitter correction.

Since the jitter effects vary by image line, the time delay between even and odd (or deselected) detectors will lead to slightly different jitter effects in adjacent image samples, as depicted in Figure 4-38. The figure shows six time samples (t0 through t5) for six adjacent detectors. Note that the input line location returned by the grid is adjusted differently for the even and odd detectors due to their timing offset. Including the effects of detector deselect, the interpolated line location for the hybrid pixels could be different for each detector. The current approach does not account for sample-to-sample variations in jitter for each detector, applying the jitter correction only at the output location. This preserves the uniform along-track sampling assumption required to apply the cubic convolution kernel. Also note that while the interpolation location is

adjusted relative to the input pixel locations in the line direction, the detector sample locations are adjusted relative to the interpolation location in the sample direction. The jitter-adjusted resampling procedure is explained in more detail below.

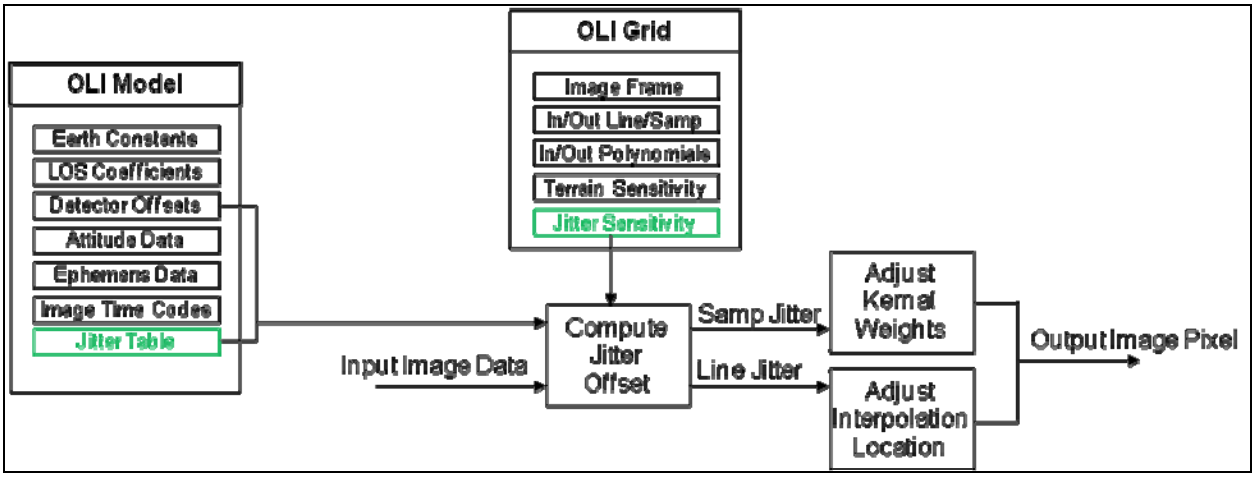

Figure 4-37. OLI LOS Model and OLI LOS Grid Jitter Correction Data Flow

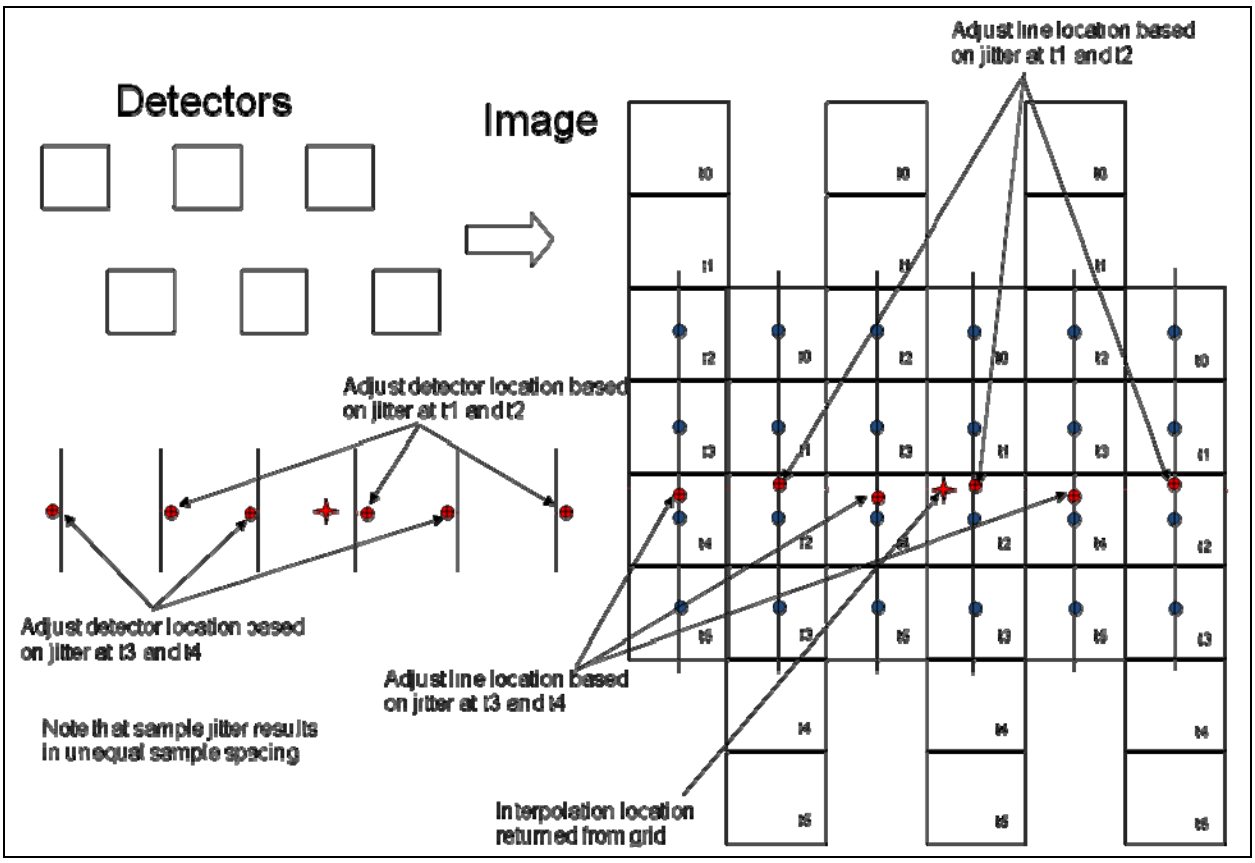

Figure 4-38. Jitter Effects in Image Resampling

4.2.4.6.3 Building The SCA-trimmed Look-Up Table (LUT).

Allocate the SCA-trim LUT. There is a starting and ending sample location of active or valid imagery stored for each line of output in the SCA-trimming look-up table.

LUT = malloc(2 * nl)

Where nl = number of lines in output imagery

Given the set of geographic corner coordinates, read from the input grid file, that represent valid imagery for a given band:

1. Map four corners to the output projection coordinates.
2. Map four output projection coordinates to line and sample coordinates.
3. Set up polygon definition from four coordinates:
   <px0,py0> = <sample upper left, line upper left>
   <px1,py1> = <sample upper right, line upper right>
   <px2,py2> = <sample lower right, line lower right>
   <px3,py3> = <sample lower left, line lower left>
   <px4,py4> = <sample upper left, line upper left>
4. Set up sample locations for each line that is outside of active imagery:
   osamp1 = -1.0
   osamp2 = output number of samples
   for nn = 0 to 3
   if px[nn] < osamp1 then osamp1 = px[nn] – 1.0
   if px[nn] > osamp2 then osamp2 = px[nn] + 1.0
5. Initialize LUT values to fill for all output lines:
   For nn = 0 to (2 * number of output lines)
   LUT[nn] = 0
6. For nn = 0 to number of output lines (nn and current line are synonymous).
   6.1. Define line by sample locations calculated from 4 and current line
   <x0,y0> = <osamp1, nn>
   <x1,y1> = <osamp2, nn>
   6.2. Determine the intersection between the sides of the polygon defined in 3 and line defined in 6.1
   Initialize the number of intersections for the current line:
   intersections = 0
   For nn = 0 to 3
   (Simple line intersection routine)
   xlk = x0 – x1
   ylk = y0 – y1
   xnm = px[nn] – px[nn+1]
   ynm = py[nn] – py[nn+1]
   xmk = px[nn+1] – x1
   ymk = py[nn -1] – y1
   det = xnm * ylk – ynm * xlk
   if (| det | <= TOL) lines are parallel, no intersection found.
   s = (xnm * ymk - ynm * xmk) / det
   t = (xlk * ymk - ylk * xmk) / det
   if(s<0.0 || s>1.0 || t<0.0 || t>1.0)

```
            no intersection found
        else
            intersection found, calculate point:
            xp[ intersections ]  = x1 + xlk * s
            yp[ intersections ] = y1 + ylk * s
            intersections++
```

6.3. If the number of intersections from 6.2. is two, then the current line has valid active imagery and the look up table values are these intersections and represent the start and stop of valid imagery. Store values in the SCA-trim lookup table.

```
        if   xp[0] > xp[1]
            LUT[ 2 * nn ]      = xp[1]
            LUT[ 2 * nn + 1] = xp[0]
        else
            LUT[ 2 * nn ]      = xp[0]
            LUT[ 2 * nn + 1] = xp[1]
```

(Note: If the number of intersections is not two, then the current line has no valid active imagery and the SCA-trim lookup table will contain points outside of the imagery, and fill will be used).

4.2.4.6.4 Load/Build Information

To resample a Level 1R dataset, the image file, grid file, geometric model (and, if the effects of terrain are to be removed, a DEM) must be opened.

4.2.4.6.5 Resample Level1R Imagery

Loop on each band of each SCA for resampling.

1. Get the resampling grid for the band and SCA to be processed.
2. Build the SCA-trimming table.
3. Read one band of imagery for one SCA.
   3.1. Initialize the jitter correction parameters.
   If the current band is panchromatic, then jitter_scale = 1
   Otherwise, jitter_scale = 2
4. Loop on output line/samples.
   4.1. Check if the output line/sample is within SCA-trimming bounds.

```
   if   output sample > LUT[ 2 * output line ] &&
            output sample < LUT[ 2 * output line + 1 ]  then proceed
   else       output pixel = fill
```

   4.2. If the image is terrain corrected, calculate the elevation-dependent input line/sample location.
   4.2.1. Get the elevation for the output pixel location X/Y location from DEM (elevation).
   4.2.2. Map the output line/sample back into the input space, using the grid and the function 3d_ols2ils.
   4.3. If the image is not terrain corrected, calculate the zero elevation (ellipsoid surface) input line/sample location.
   4.3.1. Set the elevation to zero.

4.3.2. Map the output line/sample back into the input space, using the grid and the function ols2ils.

4.4. Calculate the actual input sample location; for sample location (int)input sample calculated from either 4.2 or 4.3:

4.4.1. Calculate the detector offset parallax scale.

Scale = (int) floor(detector along-track offset + 0.5) (in the geometric model).

4.4.2. Calculate the sample odd/even parallax offset.

$\Delta sample\_oe = (d_0 + elevation * d_1) * scale$

Note that $(d_0 + elevation * d_1)$ is the parallax (in pixels) per pixel of along-track offset from the nominal detector location.

Where

$d_{0,1}$ = odd/even sample parallax coefficients stored in the grid

4.4.3. Get the sample fractional offset

fractional sample offset =

detector across-track offset (in the geometric model)

4.4.4. Calculate the sample jitter adjustment.

4.4.4.1. Calculate the index into the jitter table for the current image line

jit_index = (int)(jitter_scale*(input line – pixel column fill (defined below)))

Make sure the jitter index is within the range of the jitter table. Set to the min or max value (whichever is closest) if it is outside the range.

4.4.4.2. Calculate the fractional jitter table index.

$\Delta jit\_index = jitter\_scale * input\ line - floor(jitter\_scale * input\ line)$

4.4.4.3. Calculate the simple sample jitter adjustment.

```
samp_jitter0 = samp_sens[0] * jitter_table[jit_index].roll
             + samp_sens[1] * jitter_table[jit_index].pitch
             + samp_sens[2] * jitter_table[jit_index].yaw
samp_jitter1 = samp_sens[0] * jitter_table[jit_index+1].roll
             + samp_sens[1] * jitter_table[jit_index+1].pitch
             + samp_sens[2] * jitter_table[jit_index+1].yaw
```

$samp\_jitter = samp\_jitter0 * (1-\Delta jit\_index) + samp\_jitter1 * \Delta jit\_index$

Where:

samp_sens[0] is the sample direction jitter roll sensitivity,
samp_sens[1] is the sample direction jitter pitch sensitivity,
samp_sens[2] is the sample direction jitter yaw sensitivity,
for the current grid cell, from the OLI grid.
jitter_table[n] is the jitter table roll-pitch-yaw vector for row n,
from the OLI model.

4.4.4.4. Refine the sample jitter to compensate for line jitter.

```
line_jitter0 = line_sens[0] * jitter_table[jit_index].roll
             + line_sens[1] * jitter_table[jit_index].pitch
             + line_sens[2] * jitter_table[jit_index].yaw
line_jitter1 = line_sens[0] * jitter_table[jit_index+1].roll
             + line_sens[1] * jitter_table[jit_index+1].pitch
             + line_sens[2] * jitter_table[jit_index+1].yaw
```

$line\_jitter = line\_jitter0 * (1-\Delta jit\_index) + line\_jitter1 * \Delta jit\_index$

Where:

line_sens[0] is the line direction jitter roll sensitivity,
line_sens[1] is the line direction jitter pitch sensitivity,
line_sens[2] is the line direction jitter yaw sensitivity,
for the current grid cell, from the OLI grid.

This is the error in the line coordinate used above, due to line jitter.

samp_rate =
samp_sens[0]*(jitter_table[jit_index+1].roll-jitter_table[jit_index].roll)
+ samp_sens[1]*(jitter_table[jit_index+1].pitch-
jitter_table[jit_index].pitch)
+ samp_sens[2]*(jitter_table[jit_index+1].yaw-jitter_table[jit_index].yaw)

This is the rate of change of sample jitter with the line coordinate.

samp_jitter += line_jitter*samp_rate

This is the sample jitter correction adjusted for the effects of line jitter.

4.4.4.5. actual input sample = input sample - Δsample_oe - samp_jitter - fractional sample. These corrections are subtracted rather than added because, rather than adjusting the input space interpolation location, we are computing the apparent location of the detector to the left of the interpolation location to make sure we have the correct range of samples to feed the interpolation logic. If the above adjustments lead to the “actual input sample” being greater than (to the right of) the original input sample location, then we move our sample range one more sample to the left. We perform a similar calculation on the detector to the right of the input space interpolation location to make sure that we do not have to shift one more sample in that direction. See also the note in 4.6.2 below.

4.5. Create the fractional pixel shift for the current input location:

Δline = input line - (int) input line

Δsample = input sample - (int) input sample

4.6. Create aligned samples for Akima resampling by applying cubic convolution weights in line direction.

4.6.1. Loop on the actual input sample location:

For hybrid sample = (int) actual input sample - 2 to (int) actual input sample +3. One extra hybrid sample created to the left and right of the minimum number of samples needed for Akima interpolation.

In the case of NN resampling, the loop limits are reduced to:

For hybrid sample = (int) actual input sample to (int) actual input sample +1

4.6.1.1. Calculate the line and hybrid sample detector offset parallax scale scale = (int) floor(detector along-track offset + 0.5) (in the geometric model).

4.6.1.2. Calculate the odd/even detector offset, parallax correction, and jitter correction for the hybrid detector.

4.6.1.2.1. Odd/even detector offset and parallax corrections.

$\Delta line\_oe = (c_0 + elevation * c_1) * scale + pixel\ column\ fill - nominal\ detector\ fill - at\_offset[hybrid\ sample]$

$\Delta sample\_oe = (d_0 + elevation * d_1) * scale$

Where:

$c_{0,1}$ = odd/even line parallax coefficients stored in the grid
$d_{0,1}$ = odd/even sample parallax coefficients stored in the grid. Note that (c_0 + elevation * c_1) is the along-track parallax (in pixels) per pixel of along-track offset from the nominal detector location and (d_0 + elevation * d_1) is the across-track parallax (in pixels) per pixel of along-track offset from the nominal detector location.

4.6.1.2.2. Jitter correction

The sample jitter correction is calculated as described in 4.4.4 above. The line jitter correction is calculated as follows:

jit_index = (int)(jitter_scale*(input line – pixel column fill))
Δjit_index = jitter_scale * input line – floor(jitter_scale * input line)
line_jitter0 = line_sens[0] * jitter_table[jit_index].roll
+ line_sens[1] * jitter_table[jit_index].pitch
+ line_sens[2] * jitter_table[jit_index].yaw
line_jitter1 = line_sens[0] * jitter_table[jit_index+1].roll
+ line_sens[1] * jitter_table[jit_index+1].pitch
+ line_sens[2] * jitter_table[jit_index+1].yaw
line_jitter = line_jitter0 * (1-Δjit_index) + line_jitter1*Δjit_index
Where:
line_sens[0] is the line direction jitter roll sensitivity,
line_sens[1] is the line direction jitter pitch sensitivity,
line_sens[2] is the line direction jitter yaw sensitivity,
for the current grid cell, from the OLI grid.
This is the error in the line coordinate due to jitter.
line_rate =
line_sens[0]*(jitter_table[jit_index+1].roll-jitter_table[jit_index].roll)
+ line_sens[1]*(jitter_table[jit_index+1].pitch-jitter_table[jit_index].pitch)
+ line_sens[2]*(jitter_table[jit_index+1].yaw-jitter_table[jit_index].yaw)
This is the rate of change of line jitter with the line coordinate.
line_jitter += line_jitter*line_rate
This is the line jitter correction adjusted for the second order effects of line jitter. Note the similarity to the sample correction described in 4.4.4.4.

4.6.1.3. Calculate the new hybrid line location.

4.6.1.3.1. hybrid line = (int)floor(input line + Δline_oe + line_jitter) .
Note that in this case, we add the corrections since we are adjusting the interpolation location.

4.6.1.4. Calculate the new fractional hybrid line location.

Δhybrid line = input line + Δline_oe +line_jitter – hybrid line
If |Δhybrid line| > 1, then the integer line index must be adjusted and Δhybrid line brought back into the -1 < Δhybrid line < 1 range.

4.6.1.5. Apply cubic convolution in the line direction to hybrid sample line DNs.

4.6.1.5.1. Calculate the cubic convolution weights.

$$w_{n+2} = \sum_{n=-1}^{2} f(n - \Delta\text{hybrid line})$$

Where *f* is equal to the cubic convolution function.

4.6.1.5.2. Apply the cubic convolution weights to L1R DNs.

hybrid line DN = $w_0 * h_0 + w_1 * h_1 + w_2 * h_2 + w_3 * h_3$

Where

w_0, w_1, w_2, w_3 = Cubic convolution weights for Δhybrid line.

h_0 = DN from L1R for hybrid sample, input line location - 1

h_1 = DN from L1R for hybrid sample, input line location

h_2 = DN from the L1R for the hybrid sample, input line location + 1

h_3 = DN from the L1R for the hybrid sample, input line location + 2

In the case of NN resampling, the values of the hybrid line and Δhybrid line are used to select the closest line for the current detector/sample column, instead of being used to compute weights. The hybrid line DN is then the L1R DN value for the closest line location.

4.6.2. Calculate the apparent Akima pixel location for the current hybrid sample.

Akima pixel location x_i =

hybrid sample location - Δsample_oe

- across-track detector offset (in geometric model)

- samp_jitter (computed per 4.6.1.2.2 above)

Note that in this case, the across-track terrain parallax and sample jitter effects are subtracted instead of added. This is because we are adjusting the apparent detector location relative to the output pixel interpolation point instead of adjusting the output pixel interpolation location itself. We must do it this way in the sample direction because the adjustments are different for each detector. As for the across-track offset term, which is also unique for each detector, the detector offset corrections are designed to be applied as line-of-sight corrections in the instrument coordinate system. As such, the along-track offset is a +X LOS correction and the across-track offset is a +Y LOS correction. The instrument +X axis is in the +line direction, but the +Y axis is in the –sample direction, so this correction is also subtracted from the apparent detector location.

4.7. Calculate the output DN using Akima interpolation and hybrid line/sample information from 4.6.1 and 4.6.2.

4.7.1. Calculate the Akima weights according to the pixel locations from 4.6.2.

$$m_0 = \frac{\mathrm{DN}_1 - \mathrm{DN}_0}{x_1 - x_0}$$

$$m_1 = \frac{\mathrm{DN}_2 - \mathrm{DN}_1}{x_2 - x_1}$$

$$m_2 = \frac{\mathrm{DN}_3 - \mathrm{DN}_2}{x_3 - x_2}$$

$$m_3 = \frac{\mathrm{DN}_4 - \mathrm{DN}_3}{x_4 - x_3}$$

$$m_4 = \frac{\mathrm{DN}_5 - \mathrm{DN}_4}{x_5 - x_4}$$

$$ak_0 = \mathrm{DN}_2$$

$$ak_1 = \frac{\left|m_3 - m_2\right| * m_1 + \left|m_1 - m_0\right| * m_2}{\left|m_3 - m_2\right| + \left|m_1 - m_0\right|}$$

$$ak_2 = \frac{(3.0 * m_2 - 2.0 * ak_1 - \frac{\left|m_4 - m_3\right| * m_2 + \left|m_2 - m_1\right| * m_3}{\left|m_4 - m_3\right| + \left|m_2 - m_1\right|}}{x_3 - x_2}$$

$$ak_3 = \frac{ak_1 + \frac{\left|m_4 - m_3\right| * m_2 + \left|m_2 - m_1\right| * m_3}{\left|m_4 - m_3\right| + \left|m_2 - m_1\right|} - 2.0 * m_2}{(x_3 - x_2)^2}$$

Where:
DN_n = hybrid DNs calculated from cubic convolution, step 4.6.1.
x_n = Akima locations calculated in step 4.6.2.
ak_n = Akima weights
m_n = Akima slopes

4.7.2. Calculate the output pixel DN using the Akima A method.

$$\mathrm{outputDN} = ak_0 + ak_1 * ds + ak_3 * ds^2 + ak_4 * ds^3$$

Where

$$ds = (\Delta\mathrm{sample} + x_2)$$

The output sample point is located between hybrid samples x_2 and x_3, where x_n is from n=0…5.
In the case of NN resampling, the Akima pixel locations for the two closest detectors are examined to see which is closest to the output location. The hybrid line DN value for the closest detector is selected as the output DN value.

4.8. Write the output DN to the image file. .

5. Write out the data descriptor record for the image file. Table 4-14 shows the baseline contents of the data descriptor record. All fields present in the table refer to the imagery associated with the DDR, unless otherwise specified. Note that the scene roll angle is a new field added for off-nadir acquisitions. It would be computed from the LOS model by interpolating the roll angle from the "original" attitude data sequence at the time corresponding to the precision model reference time t_ref. This would be done using the logic described in the Find Attitude sub-algorithm in the LOS Projection Algorithm (4.2.2), except operating on the "original" rather than the "corrected" attitude data sequence. The logic for using the "original" data is so that this scene roll value will not change due to LOS model correction. The sign convention on the roll angle is such that a positive roll angle would correspond to a positive orbital Y coordinate that is looking to starboard.

4.2.4.6.6 Combining SCAs into one output file.

For an SCA combined output image, the overlap region between SCAs can be handled by averaging the pixels between SCAs.

Input and Output File Details

Output is an L1G image file formatted according to LSDS 1822 Landsat 8 (L8) Level 1 (L1) Data Format Control Book (DFCB). The output is an HDF5 file. Table 4-14 list the metadata associated with the output file. This table follows the metadata fields in the L8 L1 DFCB. The metadata is split up into a file metadata and band metadata. For further information on this format, see the L8 L1 DFCB.

Field	Description	Type
Spacecraft source	*Spacecraft associated with data record*	*char[32]*
Instrument source	*Imaging instrument (TIRS)*	*char[32]*
WRS path	WRS path number	integer
WRS row	WRS row number	integer
Capture direction	*Ascending or descending pass*	*char[64]*
Capture date	*Date imagery was acquired by instrument*	*char[11]*
Capture time	*Time of day imagery was acquired by instrument*	*char[8]*
Scene roll angle	*Roll angle (in degrees) at the scene center*	*float*
Correction type	*Raw, L1R, L1G, L1Gt, L1T*	*char[8]*
Acquisition type	*Earth, lunar, stellar*	*char[8]*
Projection Code	Map projection code	integer
Zone code	UTM zone code	integer
Datum code	Datum code for map projection	char[16]
Spheroid code	Earth model for map projection	integer
Projection units	Distance units	char[8]
Projection coefficients	Parameters needed by coordinate transformation package.	float[16]
Resampling Type	Resampling method (CC, NN)	char[4]
Software Version	Software version used to create image	char[11]
Sun Azimuth	Sun Azimuth at scene center	float
Sun Elevation	Sun Elevation at scene center	float
Sun Angles Valid	1 if sun data present 0 if not	char
Sun Angle Correction	None, scene center, per-pixel	char[32]
Earth Sun Distance	Earth sun distance	float
For each band:		
Band Number	Landsat 8/9 band designation for current record	integer
Number lines	Number of lines present in data file	integer
Number samples	Number of samples present in data file	integer
Data Type	Data type of imagery	integer
Maximum Pixel	Maximum DN value in data	float
Minimum Pixel	Minimum DN value in data	float
Maximum Radiance	Maximum radiance	float
Minimum Radiance	Minimum radiance	float
Upper left projection coordinate	Upper-left y (latitude/northing) and x (longitude/easting) coordinate	float[2]
Upper right projection coordinate	Upper-right y (latitude/northing) and x (longitude/easting) coordinate	float[2]
Lower right projection coordinate	Lower-right y (latitude/northing) and x (longitude/easting) coordinate	float[2]
Lower left projection coordinate	Lower-left y (latitude/northing) and x (longitude/easting) coordinate	float[2]
Projection distance y	Pixel size for y map coordinate	float
Projection distance x	Pixel size for x map coordinate	float

Field	Description	Type
Maximum Pixel Value	Maximum DN of pixels	float
Minimum Pixel Value	Minimum DN of pixels	float
Pixel range valid	Indicates valid max / min is listed	char
Maximum Radiance	Maximum radiance value	float
Minimum Radiance	Minimum radiance value	float
Spectral Radiance Scaling Offset	Offset to convert to radiance	float
Spectral Radiance Scaling Gain	Gain to convert to radiance	float
Radiance valid	Indicates valid radiance items are present	char
Instrument Source	Source of data	char[32]

Table 4-14. L1G File Metadata Fields

4.2.5 Terrain Occlusion Mask Generation Algorithm

4.2.5.1 Background/Introduction

The heritage Landsat and ALI/EO-1 image resampling procedures ignored the possibility of multiple terrain intersections due to off-nadir viewing toward the edges of the imaging swath. This was a reasonable simplification for Landsat with its fixed nadir viewing geometry. Although the ALI was capable of off-nadir pointing, this capability was mostly used to acquire different portions of the nominal Landsat swath, given that the ALI's focal plane was only 20 percent populated. Furthermore, EO-1 was a technology demonstration project with a minimal budget for ground processing algorithm development, so Landsat capabilities were reused as-is wherever possible.

Ignoring the multiple terrain intersection effect is less defensible for Landsat 8/9, which will routinely acquire off-nadir scenes from adjacent WRS paths for product generation. The approach to this problem adopted here is to compute the ground locations where the OLI line-of-sight is obstructed by terrain, and provide this information in a mask. The image resampling logic will be permitted to populate all output image pixels with apparent values according to the heritage algorithm. Some of these will be erroneous data that actually represent terrain intersection points closer to the imaging sensor. These can be subsequently identified, and if appropriate, replaced with fill by the user, based on the contents of the terrain occlusion mask generated by this algorithm. This approach was felt to be preferable to inserting fill in the product image, as some image exploitation algorithms (e.g., control point mensuration) are sensitive to the presence of fill.

Generating the terrain occlusion mask can also be performed without reference to the output image itself, requiring only the DEM and the LOS projection grid as inputs. For each pixel in the output image, the algorithm uses the grid file to locate the corresponding pixel in the input (L1R) space. It then uses the grid to compute the output space line/sample location corresponding to the same input line/sample at the maximum elevation plane. The line connecting the original output pixel location with the maximum elevation location corresponds to the projection of that pixel's line-of-sight into output space. By interpolating elevation model heights for points along this line and

comparing them to the computed LOS height, terrain intersection points that are closer to the imager can be detected. Each point in the output terrain occlusion mask will be flagged as either visible or occluded by terrain.

4.2.5.2 Dependencies

The terrain occlusion algorithm assumes that the LOS Projection and Gridding Algorithm (4.2.2) has created the output product LOS projection grid. The elevation planes in the LOS projection grid must span the range of elevations in the elevation model.

4.2.5.3 Inputs

The terrain occlusion algorithm and its component sub-algorithms use the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs
ODL file (implementation)
OLI Grid file
DEM Grid file
Original Unresampled DEM file
Terrain Occlusion Mask file name
Terrain Occlusion band

4.2.5.4 Outputs

TO (terrain occlusion) mask file
TO mask data descriptor record (DDR) (see note 4)
TO mask image

4.2.5.5 Options

None.

4.2.5.6 Procedure

Read the unresampled DEM to determine the maximum elevation within the file (maximum_elevation).

Initialize the terrain mask to 0.

For each SCA:

For each output pixel:

1. Retrieve the elevation for the current output pixel location (current elevation) from the DEM.
    a. Using the DEM resampling grid, map the L1TP output pixel location to geographic unresampled DEM line/sample location.
        i. Calculate grid cell row and column index.

grid row = output line / number grid cell lines

grid col = output sample / number grid cell samples

ii. Determine grid cell number.

grid cell number = grid row * number grid cell samples +
grid col

iii. Look up grid mapping coefficients based on grid cell.

coeff = grid cell coefficient reverse[grid cell number].

iv. Calculate DEM line/sample location.

DEM line = coeff.line[0] +

output sample * coeff.line[1] +

output line * coeff.line[2] +

output sample * output line * coeff[3].line

DEM sample = coeff.sample[0] +

output sample * coeff.sample[1] +

output line * coeff.sample[2] +

output sample * output line * coeff[3].sample

b. Perform bilinear interpolation at location in DEM from step 1a) to determine elevation of current L1TP output location.

i. Determine subpixel location

Integer line = (int)DEM line

Integer sample = (int) DEM sample

ds = DEM sample – Integer sample

dl = DEM line – Integer line

ii. Determine location in DEM image buffer.

dem_ns = number samples in DEM

dem_nl = number lines in DEM

loc = Integer line * dem_ns + Integer Sample

iii. Interpolate elevation for floating point location.

elevation =

(1.0 - ds) * (1.0 - dl) * dem.data[loc] +

ds * (1.0 - dl) *dem.data[loc+1] +

(1.0 - ds) *dl * dem.data[loc+dem_ns] +
ds * dl * dem->data[loc+dem_ns + 1]

Note:
For off-nadir images, pixel line-of-sight ground projections can extend outside of the product image area. Using the unresampled DEM as the source of elevation data should prevent elevations from being needed outside of the available data range as the terrain occlusion calculation performs its "stepping process." However, a check to ensure that the elevation being retrieved is greater than 0 in line and sample while less than dem_nl-1 and dem_ns-1 should be implemented. The process should issue a warning that the data to be retrieved is outside of the DEM, and return the DEM elevation value for the closest edge line/sample position (i.e., clip the DEM line/sample values at the DEM edges).

2. Run ols2ils (OLI Resampling Algorithm Algorithm 4.2.4) to find the input location for the corresponding output location. This will be based on the elevation for current output pixels (l_c,s_c).
3. For the input location calculated in 2) calculate the corresponding output location for the maximum elevation (l_m,s_m) (OLI Resampling Algorithm Algorithm 4.2.4).
4. Define the parametric equation for a line that connects (l_c,s_c) to (l_m,s_m).

   $s_p = s_0 + t * f$

   $l_p = l_0 + t * g$

   where: $0 \le t \le 1$

   At t=0: $l_p=l_c$ and $s_p=s_c$.

   At t=1: $l_p=l_m$ and $s_p=s_m$

   Therefore

   $l_0 = l_c$,

   $s_0=s_c$,

   $g=(l_m-l_c)$,

   $f=(s_m-s_c)$

5. Compute the length of the line in output space:

   $$d = MAX(1, \sqrt{(s_m - s_c)^2 + (l_m - l_c)^2}$$

6. Compute the increment of t to use to walk along the line:

   $$\Delta t = \frac{MAX\left(|s_m - s_c|, |l_m - l_c|, 1\right)}{d^2}$$

7. Walk along the line in increments of Δt, testing each point for terrain occlusion:

   For j = 0 to (int)ceil(1/Δt)

   t = j * Δt

8. Calculate the point of intersection:

   $l_p = l_0 + t * g$

$s_p = s_0 + t * f$

9. Round (l_p,s_p) to get (l_p',s_p'). Find the elevation for (l_p',s_p') (pixel elevation) using the DEM resampling grid as described in steps 1a) and 1b) above.
10. The value of t represents the ratio used to measure whether the elevation of (l_p',s_p') is large enough to obscure the current pixel of interest (l_c,s_c).

    if((t * maximum elevation + (1.0-t) * current elevation) < pixel elevation)

    Current pixel location (l_c,s_c) is occluded. Set the terrain mask to 1 and exit loop.

    else

    Current pixel location (l_c,s_c) is not occluded. Continue to loop.

Determining Elevation (change from using co-registered DEM)
Due to the "walk-a-line" process of step 7) of the previous procedure, the location of an elevation requested could reside outside of the co-registered DEM used in creating the L1TP. To account for this, the unresampled DEM and DEM geomgrid (used to resample the DEM) are used for terrain occlusion calculations instead of the co-registered DEM. Since the unresampled DEM should extend outside the boundary of the L1TP, this will allow the retrieval of elevations outside the product image extent.

4.2.5.7 Notes

Some additional background assumptions and notes include the following:

1. The new logic required to calculate the terrain occlusion mask (particularly for off-nadir scenes) is documented here, but may be implemented as part of the resampling software for processing efficiency. The Terrain Occlusion (TO) mask output by this algorithm is also included as a possible (to be resolved) output in the resampling algorithm.
2. The current concept is to allow the user to specify the band(s) to use in testing for occlusion. However, for the terrain mask that is to accompany the L1TP L8/9 product, generation of the mask for the SWIR1 band should be sufficient.
3. The problem of multiple terrain intersections needs to be addressed, particularly for off-nadir acquisitions. A terrain occlusion mask will be generated to identify these obstructed pixels, but the current thinking is that it would not alter the behavior of the resampler, as sprinkling fill pixels throughout a product image can wreak havoc with some applications. Generating a separate terrain occlusion mask will allow users to evaluate the extent of the problem and apply the mask if appropriate to a particular application.

4.2.6 OLI Geometric Accuracy Assessment (L1TP)

4.2.6.1 Background/Introduction

The OLI geometric accuracy assessment, or geometric characterization, algorithm analyzes the results of the GCP measurements derived through correlation of the GCP image chips with a precision- and terrain-corrected OLI L1TP image. Unlike the similar

geodetic accuracy assessment algorithm (6.2.7), there is no precision LOS model correction step invoked to analyze the GCP results and detect outliers. Instead, the geometric accuracy assessment is applied directly to the results of control point mensuration in the L1TP image. Statistics are computed for the GCP measurements, with outliers detected and rejected based upon a t-distribution test. The GCP results provide a measure of the accuracy of the output L1TP product through direct comparison to an absolute ground control source. Ideally, a different set of GCPs would be used for geometric accuracy assessment than were used in the L1TP LOS model correction process. This will require flagging some GCPs in the GCP library as validation points to be withheld from the original GCP mensuration and LOS model correction process and used only for geometric characterization. Setting these control/validation flags is a Cal/Val Team responsibility.

4.2.6.2 Dependencies

The OLI geometric accuracy assessment algorithm assumes that the L1TP product generation flow has been executed to create an L1TP image, and that this image has been correlated with a set of validation GCP image chips (see note #3 in Section 4.2.6.7), using the GCP correlation algorithm, to produce a set of GCP measurements. Normally, this L1TP image will be a standard SCA-combined L1TP product, but the GCP correlation algorithm may also be run on SCA-separated images in testing and anomaly resolution scenarios.

4.2.6.3 Inputs

The geometric accuracy characterization algorithm uses the inputs listed below. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs
ODL file
Measured GCP File Name
Band to process (See note 6)
L1TP Image File Name (for access to metadata)
T-distribution outlier threshold
Output geometric report file name
L0Rp ID (for trending)
Work Order ID (for trending)
Characterization Database Output (Trending) Flag (on/off)
GCP residual output (on/off)
Measured GCP File Contents (see GCP Correlation Algorithm for details)
GCP point ID
GCP latitude and longitude (in degrees) and height (in meters)
Correlation success/failure flag
Predicted GCP image line and sample location
Line and sample offsets in pixels
Measured band number
GCP source (GLS or DOQ)
L1TP Image File Metadata Contents (see Image Resampling Algorithm for details)
WRS Path/Row (for trending)

Image pixel size (for measured band) in meters
Acquisition date (for trending)
Acquisition type (for standard report file header)
Scene roll angle (for trending and report file)

4.2.6.4 Outputs

Geometric Accuracy Report (output file and trending) (see Table 4-15 below for additional details)
Processing Information
Processing Date and Time
Processing Center/Location
Processing Software Version
Processed L1TP Image File Name
Dataset Information
Spacecraft and Instrument Source (Landsat 8/9 and OLI/OLI-2)
Work Order ID
WRS Path/Row
Off-Nadir Angle
Acquisition Type (Earth, Lunar, Stellar) (will always be Earth)
L0Rp ID
Acquisition Date
GCP Information
GCP Source
Pixel Size (meters)
Number of valid GCPs
Mean latitude and longitude of GCPs
GCP Statistics
Mean, RMSE, Standard Deviation, Correlation Coefficient
GCP Residual Output Fields (written to report file only if option is selected)
For each valid GCP:
GCP point ID
GCP latitude and longitude (in degrees) and height (in meters)
Line and sample offsets scaled to meters
Measured band number
Measured SCA number (usually 0) (see note #4)

4.2.6.5 Options

Characterization database output on or off.
GCP residual output on or off.

4.2.6.6 Procedure

Geometric characterization is performed on the measured GCP file output created by the GCP correlation algorithm. See the GCP Correlation Algorithm in Section 4.1.6 for further details on the GCP mensuration process and its results. Geometric characterization reads the entire set of GCP measurements, removes those flagged as correlation failures, performs a Student t-distribution test to detect and eliminate outliers, and calculates statistics for the remaining valid residuals. The resulting statistics reflect the accuracy of the measured L1TP image product relative to the set of validation GCPs

used in the correlation process, providing a measure of L1TP product accuracy that is independent of the control used to create the product.

The geometric accuracy assessment algorithm computes the same summary statistics as the geodetic accuracy assessment algorithm. It differs from that algorithm in that it must read the raw GCP measurement file produced by the GCP correlation algorithm instead of the precision correction residuals file produced by the LOS model correction algorithm, and therefore must filter its input for outliers prior to computing the statistics. In addition, unlike geodetic characterization, geometric characterization includes an option to write out the measured offsets for those GCP declared as valid by the outlier test. This makes it possible to capture the outlier test results for individual GCPs for further analysis.

Figure 4-39 shows the architecture for the Geometric Characterization algorithm.

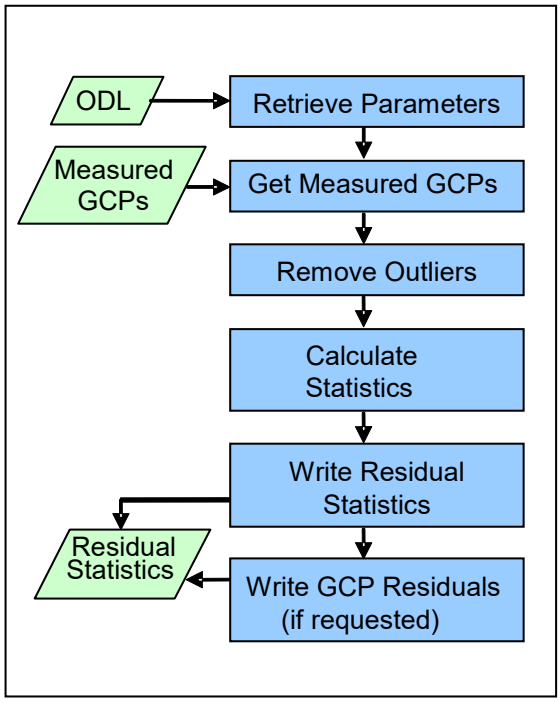

Figure 4-39. Geometric Characterization Algorithm Architecture

The geometric accuracy assessment algorithm consists of the following three stages:

Stage 1 - Read processing parameters and load GCP data.
Stage 2 - Use the Student t-distribution test to perform outlier filtering on the GCPs.
Stage 3 - Compute the mean, standard deviation, RMSE, and correlation coefficient statistics for the valid GCPs, and generate the report file and characterization database outputs.

In practice, stages 2 and 3 are somewhat overlapped, as it is necessary to compute mean and standard deviation statistics on the GCP set as part of the t-distribution outlier test.

4.2.6.6.1 Stage 1 - Load Processing Parameters and GCP Measurements

Stage 1 processing reads the processing parameters from the input ODL file and loads the GCP data from the measured GCP file.

Get Geometric Characterization Parameters Sub-Algorithm
Reads the parameters for geometric characterization from the input ODL parameter file. In addition, reads the Level 1TP image metadata to retrieve the WRS path/row, acquisition date, and pixel size for all bands in the image.

Get GCP Measurements Sub-Algorithm
This function reads the latitude, longitude, band number, GCP source, and correlation status flags, as well as the along- and across-track offset measurements for each GCP from the measurement file created by the GCP correlation algorithm. It retrieves all of the measured GCPs for subsequent correlation flag and outlier filtering. Each GCP record is loaded into a data structure containing the fields listed in Table 4-16 below.

The measured GCP line and sample offsets are scaled to units of meters using the pixel size read from the L1TP metadata for the measured band. The fields required for stage 2 and 3 processing are as follows:

1. Point ID
2. GCP Latitude
3. GCP Longitude
4. GCP Height
5. Line Offset Scaled to Meters
6. Sample Offset Scaled to Meters
7. Valid GCP (Correlation and Outlier) Flag
8. L1TP Band Number
9. L1TP SCA Number (usually 0 for SCA-combined images) (see note #4)
10. GCP Source

4.2.6.6.2 Stage 2 - Filter GCP Outliers

Stage 2 processing identifies those GCP measurements classified as outliers. The outlier set is initially identified as those GCPs flagged as image correlation failures. A statistical t-distribution outlier test is applied iteratively to the remaining valid GCPs, removing any newly identified outliers at the end of each iteration, until no new outliers are found.

Remove GCP Outliers Sub-Algorithm
This function removes the GCP records flagged as outliers from the valid GCP set. Records with the correlation flag field set to 0 are outliers. The initial valid GCP set includes the GCPs that were flagged as successful correlations. The Student t-

distribution outlier test is then performed to identify additional outlier GCPs based on the magnitude of their offsets relative to the mean offsets for the entire valid set.

Student-T Outlier Test Sub-Algorithm

Given a tolerance value, outliers are removed within the dataset until all values deemed as “non-outliers” or “valid” fall inside the confidence interval of a T-distribution. The tolerance, or probability threshold for the associated confidence interval, is specified per run and usually lies between 0.9-0.99. The default value is 0.95. The number of degrees of freedom of the dataset is equal to the number of valid data points minus one.

The steps involved in this outlier procedure are as follows:

Calculate the mean and standard deviation of the valid GCP dataset for both the line and sample directions.
Find the largest offset and compare it to the outlier threshold.

a. Find the two-tailed T-distribution (*T*) value for the current degree of freedom and confidence level specified α. See the T-Distribution Confidence Interval Computation discussion below for details on how the T-distribution confidence interval limit value (*T*) is computed based on the specified probability threshold (α). Note that this must be recomputed for each iteration since the number of degrees of freedom changes as outliers are removed.
b. Calculate largest deviation from the mean allowable for the specified degree of freedom and α:
   $\Delta line = \sigma_{line} * T$
   $\Delta sample = \sigma_{sample} * T$
   Where:
   σ_{line} = standard deviation of valid line offsets
   σ_{sample} = standard deviation of valid sample offsets
c. Find valid data point that is farthest from the mean.
   $max\ line_i = MAX\{ ABS(line\ offset - mean\ line\ offset) \}$
   $max\ sample_j = MAX\{ ABS(sample\ offset - mean\ sample\ offset) \}$
   Where:
   The maximum is found from all valid offsets
   i is the tie-point number of max line
   j is the tie-point number of max sample
d. If valid data point that is farthest from the mean is greater than the allowable Δ then the valid point is flagged as outlier.
   if $max\ line_i > \Delta line$ or $max\ sample_j > \Delta sample$ then
      if($max\ sample_j / \sigma_{sample} > max\ line_i / \sigma_{line}$)
         tie-point j is marked as an outlier
      else
         tie-point i is marked as an outlier
   else no outliers found

Repeat 1 and 2 above until no outliers are found.

T-Distribution Confidence Interval Computation

The probability density function (pdf) for the t-distribution is as follows:

$$f_r(t) = \frac{\Gamma(\frac{r+1}{2})}{\sqrt{\pi r}\,\Gamma(\frac{r}{2})\left(1+\frac{t^2}{r}\right)^{(r+1)/2}}$$

where: r = the number of degrees of freedom (n-1)
Γ = the gamma function

As pointed out in "Numerical Recipes in C," it is often more convenient (and safer) to compute the logarithm of the gamma function, as the gamma function values can get quite large and it is often ratios of gamma functions (as here) that are of interest. The t-distribution pdf can be reformulated as follows:

$$f_r(t) = e^{u(t)}$$

$$u(t) = \ln\Gamma(\frac{r+1}{2}) - \frac{\ln(\pi r)}{2} - \ln\Gamma(\frac{r}{2}) - \frac{r+1}{2}\ln\left(1+\frac{t^2}{r}\right)$$

where: lnΓ = the logarithm of the gamma function

"Numerical Recipes in C" provides a routine for computing lnΓ called gammln (ref. page 214 of 2nd edition). The gammln function is used in the reformulated t-distribution pdf shown above to compute the value of the pdf for a given t and degrees of freedom (r).

Using the t-distribution pdf, we compute the confidence level by numerical integration:

1. Initialize the integration, setting the integration step size to 0.001:
    a. step = 0.001
    b. sum = 0
    c. target = α/2 (half the probability threshold since we're only integrating the positive half of the distribution)
    d. t = 0
    e. delta = step*(t_pdf(t,dof) + t_pdf(t+step,dof))/2
2. Iterate the integration steps until the sum reaches the target:
    while(sum+delta < target)
    a. sum = sum + delta
    b. t = t + step
    c. delta = step*(t_pdf(t,dof) + t_pdf(t+step,dof))/2
3. Solve for the Δt value to exactly reach target, assuming the pdf is linear over the step size:
    a. a = (t_pdf(t+step,dof) - t_pdf(t,dof))/2/step
    b. b = t_pdf(t,doft)
    c. c = sum - target
    d. if (|a| > 0) Δt = (-b + sqrt(b^2 - 4ac))/2/a (the quadratic formula)
    else if (|b| > 0) Δt = -c/b
    else Δt = 0
4. Compute the final T value: T = t + Δt

4.2.6.6.3 Stage 3 - Calculate GCP Statistics and Create Output

The third stage of processing calculates the summary statistics for the final valid GCP set and generates the output geometric accuracy report and characterization database outputs.

Analyze GCP Residuals Sub-Algorithm

This function calculates the mean, RMSE, and standard deviation of the along- and across-track GCP residuals, as well as the correlation coefficient between the across- and along-track residuals. The statistics are computed in the following process:

a) Calculate the GCP statistics
   - a1) Calculate the total number of GCPs used (count of valid GCPs)
   - a2) Calculate the mean latitude of GCPs used
   - a3) Calculate the mean longitude of GCPs used

b) Calculate the offset statistics
   - b1) Calculate the mean of line offsets
   - b2) Calculate the mean of sample offsets
   - b3) Calculate the RMSE of line offsets
   - b4) Calculate the RMSE of sample offsets
   - b5) Calculate the standard deviation of line offsets
   - b6) Calculate the standard deviation of sample offsets
   - b7) Calculate the correlation coefficient between line and sample offsets

The following equations are used to perform these calculations, with X being the parameter for which statistics are calculated:

Mean:

$$X_{mean} = \frac{1}{numGCP} \sum_{i=1}^{numGCP} X_i$$

RMSE:

$$X_{RMS} = \sqrt{\frac{1}{numGCP} \sum_{i=1}^{numGCP} X_i^{\,2}}$$

Standard Deviation:

$$X_{StdDev} = \sqrt{\frac{1}{numGCP - 1}\left(\left(\sum_{i=1}^{numGCP} X_i^{\,2}\right) - numGCP * X_{mean}^{\,2}\right)}$$

Correlation Coefficient:

$$XY_{CC} = \frac{\left(\sum_{i=1}^{numGCP} (X_i - X_{mean})(Y_i - Y_{mean})\right)}{(numGCP - 1)} \frac{1}{X_{StdDev} Y_{StdDev}}$$

Output Geometric Statistics Sub-Algorithm
This function creates the output geometric report file and writes the statistics computed from the GCP offsets to the output file. Note that the output of trending data to the characterization database is performed by the geometric characterization main procedure.

Write Geometric Statistics Sub-Algorithm
This function writes the standard report file header and then writes the GCP offset statistics to the ASCII output file.

Write GCP Residuals Sub-Algorithm
If the option to write the valid GCP measurements to the output report file is selected, this sub-algorithm is invoked to loop through the GCP list, writing those with the outlier/valid flag set to 1 to the output report file. The individual GCP measurements are not written to the characterization database. The following fields are written:

- Point ID
- GCP Latitude (in degrees)
- GCP Longitude (in degrees)
- GCP Height (in meters)
- Line Offset (scaled to meters)
- Sample Offset (scaled to meters)
- L1T Band Number
- L1T SCA Number (will be 0 for SCA-combined images) (see note #4)

Algorithm Output Details
Table 4-15 summarizes the geometric accuracy assessment algorithm outputs. All fields are written to the output report file (subject to the GCP residual output flag setting), but only those with "Yes" in the "Database Output" column are written to the characterization database. Note that the first eleven fields listed constitute the standard report header.

Field	Description	Database Output
Date and time	Date (day of week, month, day of month, year) and time of file creation.	Yes
Spacecraft and instrument source	Landsat 8/9 and OLI/OLI-2	Yes
Processing Center	EROS Data Center SVT (see note #5)	Yes
Work order ID	Work order ID associated with processing (blank if not applicable)	Yes
WRS path	WRS path number	Yes
WRS row	WRS row number	Yes
Software version	Software version used to create report	Yes
Off-nadir angle	Scene off-nadir roll angle (in degrees)	Yes
Acquisition type	Earth, Lunar, or Stellar (only Earth-viewing scenes are used for geometric characterization)	Yes
L0Rp ID	Input L0Rp image ID	Yes

Field	Description	Database Output
L1T image file	Name of L1TP used to measure GCPs	No
Acquisition date	Date of L1TP image acquisition	Yes
GCP source	Source of GCPs (GLS or DOQ)	Yes
Pixel size	L1TP image pixel size (for measured band) in meters	Yes
Number of valid points	Number of GCPs accepted as valid	Yes
Mean GCP latitude	Mean latitude of valid GCPs (degrees)	Yes
Mean GCP longitude	Mean longitude of valid GCPs (degrees)	Yes
Line offset mean	Mean of line offsets scaled to meters	Yes
Sample offset mean	Mean of sample offsets scaled to meters	Yes
Line offset RMSE	RMSE of line offsets scaled to meters	Yes
Sample offset RMSE	RMSE of sample offsets scaled to meters	Yes
Line offset standard deviation	Standard deviation of line offsets scaled to meters	Yes
Sample offset standard deviation	Standard deviation of sample offsets scaled to meters	Yes
Correlation coefficient	Correlation coefficient between line and sample offsets (dimensionless)	Yes
If the residual output option is selected:	For each valid GCP:	
Point ID	GCP ID (see GCP Correlation Algorithm for format)	No
GCP Latitude	GCP WGS84 latitude in degrees	No
GCP Longitude	GCP WGS84 longitude in degrees	No
GCP Height	GCP WGS84 ellipsoid height in meters	No
Line Offset	Measured line offset scaled to meters	No
Sample Offset	Measured sample offset scaled to meters	No
L1T Band Number	Band in which the GCP was measured	No
L1T SCA Number	SCA in which the GCP was measured (0 for SCA-combined images) (see note #4)	No

Table 4-15. Geometric Accuracy Assessment Output Details

Accessing the Geometric Accuracy Characterization Database

Although not part of the formal geometric accuracy assessment algorithm, some comments regarding the anticipated methods of accessing and analyzing the geometric accuracy results stored in the characterization database may assist with the design of the characterization database.

The database output from the geometric accuracy assessment algorithm will be accessed by a statistical summary analysis tool that queries the characterization database to retrieve geometric accuracy results from multiple scenes. Summary mean and RMSE statistics for the scene results will be calculated and output in a report containing a comma-delimited table of the retrieved trending results, as well as the summary statistics.

The geometric results would typically be queried by acquisition date, scene off-nadir angle, WRS path/row, and/or GCP source. The most common query would be a combination of GCP source, scene off-nadir angle, and acquisition date range, for

example, selecting all of the GLS-derived results, from nadir scenes, for a given calendar quarter:

GCP_Source = "GLS"
Off_Nadir_Angle is between -0.5 and 0.5
Acquisition_Date is between 01APR2012 and 30JUN2012

The summary mean and RMSE statistics would be calculated from the mean and RMSE results for the individual scenes returned as follows:

$$Mean_{net} = \frac{1}{numScene} \sum_{i=1}^{numScene} Mean_i$$

$$RMSE_{net} = \sqrt{\sum_{i=1}^{numScene} RMSE_i^{\,2} \Big/ numScene}$$

The query results would be formatted in a set of comma-delimited records (for ease of ingest into Microsoft Excel), one record per scene. Each record would contain all of the fields written to the characterization database (items with "Yes" in the rightmost column of Table 4-15 above). A header row containing the field names should precede the database records. Two trailer rows, one containing the summary statistic names (Net Line Offset Mean, Net Sample Offset Mean, Net Line Offset RMSE, Net Sample Offset RMSE) and the second containing the comma-delimited summary statistic values, should follow the database records.

Report File Formats

The final statistics listed in Table 4-19 below show the reporting and trending that is needed for the Geometric Accuracy Assessment. Table 4-16, Table 4-17, and Table 4-18 show the contents of intermediate files.

Table 4-16 lists the output from gcpcorrelate processing, containing original unfiltered correlation results. This file is called measured.gcp in the test data directory.

Field	Description
GCP Record Fields:	One set per GCP
Point ID	GCP ID
GCP chip line location	Line location of GCP within chip
GCP chip sample location	Sample location of GCP within chip
GCP latitude	GCP WGS84 latitude in degrees
GCP longitude	GCP WGS84 longitude in degrees
GCP height	GCP WGS84 ellipsoid height in meters
Predicted GCP image line	Predicted line location of GCP in L1G image
Predicted GCP image sample	Predicted sample location of GCP in L1G image
GCP image line offset	Measured line offset from predicted location
GCP image sample offset	Measured sample offset from predicted location
Correlation success flag	Flag 0 = correlation failure, 1 = success
Correlation coefficient	Measured correlation coefficient (new)
Search band number	L1G band number used

Search SCA number	L1G SCA where GCP was found
Chip source	GCP source (DOQ or GLS or TM6)

Table 4-16. Output From GCPCorrelate

Table 4-17 lists the output format created from the Perl script meas2char2.pl. This file is a reformatting of the gcpcorrelate output file, reformatted to match the output from the Image Registration Accuracy Assessment Algorithm 4.2.8. This format is one of the two acceptable formats to the tdist function. This file has been referenced as the Data File (.dat extension) in the Image Registration Accuracy Assessment Algorithm 4.2.8 and is called geometric.dat in the test data directory.

Field	Description
Date and time	*Date (day of week, month, day of month, year) and time of file creation.*
Spacecraft and instrument source	*Landsat 8/9 and OLI/OLI-2*
Processing System	*IAS*
Work order ID	*Work order ID associated with processing (blank if not applicable)*
WRS path/row	*WRS path and row*
Software version	*Software version used to create report*
L0R image file	*L0R image file name used to create L1TP*
Processed image file name	*Name of L1TP used to create report*
Reference bands	Reference bands used in image assessment
Search bands	Search bands used in image assessment
Heading for individual tie-points	One line of ASCII text defining individual tie-point fields.
For each tie-point:	
Tie point number	Tie-point index/number in total tie-point list
Reference line	Tie-point line location in reference image (band)
Reference sample	Tie-point sample location in reference image (band)
Reference latitude	Tie-point latitude location
Reference longitude	Tie-point longitude location
Reference elevation	Elevation of tie-point location
Search line	Tie-point line location in search image
Search sample	Tie-point sample location in search image
Delta line	Measured offset in line direction
Delta sample	Measured offset in sample direction
Outlier flag	1=Valid, 0=Outlier
Correlation Coefficient	Correlation peak coefficient from correlation process
Reference band	Reference band number
Search band	Search band number
Reference SCA	SCA number that reference window was extracted from
Search SCA	SCA number that search window was extracted from
Search image	Name of search image
Reference image	Name of reference image

Table 4-17. Data File Output from TDIST

Table 4-18 lists the file format for the correlation results with outliers flagged. This file has been referenced as the Residuals File (.res extension) in the Image Registration Accuracy Assessment Algorithm 4.2.8. This file is created from the tdist executable.

There is also a file with the same name as this Residuals File with an .out extension, which is also created from the tdist process. This file has the same format as the Residuals File, but does not have the outliers flagged.

These files are called geometric.res and geometric.out, respectively, in the test data directory. These files will have the format listed below.

Field	Description
Date and time	*Date (day of week, month, day of month, year) and time of file creation.*
Spacecraft and instrument source	*Landsat 8/9 and OLI/OLI-2*
Processing System	*IAS*
Work order ID	*Work order ID associated with processing (blank if not applicable)*
WRS path/row	*WRS path and row*
Software version	*Software version used to create report*
L0R image file	*L0R image file name used to create L1TP*
Processed image file name	*Name of L1TP used to create report*
Number of records	Total number of tie-points stored in file
Heading for individual tie-points	One line of ASCII text defining individual tie-point fields.
For each band combination	
Combination header	Number of points in combination, reference band number, search band number.
For each tie-point:	
Tie point number	*Tie-point index/number in total tie-point list*
Reference line	Tie-point line location in reference image (band)
Reference sample	Tie-point sample location in reference image (band)
Reference latitude	Tie-point latitude location
Reference longitude	Tie-point longitude location
Reference elevation	Elevation of tie-point location
Search line	Tie-point line location in search image
Search sample	Tie-point sample location in search image
Delta line	*Measured offset in line direction*
Delta sample	*Measured offset in sample direction*
Outlier flag	1=Valid, 0=Outlier
Reference band	Reference band number
Search band	*Search band number*
Reference SCA	SCA number that reference window was extracted from
Search SCA	*SCA number that search window was extracted from*
Search image	Name of search image
Reference image	Name of reference image

Table 4-18. Filtered Data Points From TDIST

Table 4-19 lists the format of the statistics file for the filtered correlation results. This file has been referenced as the Statistics File (.stat extension) in the Image Registration Accuracy Assessment Algorithm 4.2.8. This file is created from the tdist executable and is called geometric.stat in the test directory.

Field	Description
Date and time	*Date (day of week, month, day of month, year) and time of file creation.*
Spacecraft and instrument source	*Landsat 8/9 and OLI/OLI-2*
Processing System	*IAS*
Work order ID	*Work order ID associated with processing (blank if not applicable)*
WRS path/row	*WRS path and row*
Software version	*Software version used to create report*
L0R image file	*L0R image file name used to create L1TP*
Processed image file name	*Name of L1TP used to create report*
t-distribution threshold	*Threshold used in t-distribution outlier rejection*
For each band combination	
Reference band	Reference band of statistics
Search band	Search band of statistics
SCA	SCA number of search image
Total tie-points	Total number of tie-points for band
Correlated tie-points	Number of tie-points that successfully correlated for band
Valid tie-points	*Total number of valid tie-points for band after all outlier rejection has been performed*
For both line and sample direction:	All statistics are given in terms of pixels
Minimum offset	Minimum offset within all valid offsets
Mean offset	*Mean offset of all valid offsets*
Maximum offset	Maximum offset within all valid offsets
Median offset	Median offset within all valid offsets
Standard deviation	*Standard deviation of all valid offsets*
Root-mean-squared	*Root mean squared offset of all valid offsets*

Table 4-19. Statistics File from TDIST

4.2.6.7 Notes

Some additional background assumptions and notes include the following:

1) The RMSE GCP statistics capture the absolute geometric accuracy performance of the L8/9 output L1TP product.
2) The trending output from this algorithm will be accessed by a statistical summary analysis tool that queries the trending database to retrieve geometric accuracy results from multiple scenes. Summary statistics (mean, standard deviation, and RMSE) for the individual scene results will be calculated and output in a report containing a comma-delimited table of the retrieved trending results, as well as the summary statistics.
3) The GCPs in the GCP repository (part of the Infrastructure Element) will be flagged as either “control” points, to be used for LOS model correction, or “validation” points, to be used for geometric accuracy assessment. Either the utility that extracts control points from this repository or the GCP correlation algorithm will extract the desired GCP set. In either case, geometric accuracy assessment would operate on the resulting output from GCP correlation. The “control” set would contain the majority of the points. The “validation” flag would only be used in areas where more than some minimum threshold number of

GCPs is available. The CalVal Team would set these flags at the time the GCP repository was loaded and these could be adjusted, if necessary, thereafter.

4) The GCP residual output option includes writing the GCP SCA number to the output report file. Under normal conditions, this field will always be zero, indicating that GCP mensuration was performed on an SCA-combined image. For anomaly investigation and testing purposes, it may be desirable to perform GCP mensuration on an SCA-separated image. For example, to use geometric accuracy assessment to analyze the GCP correlation output for a scene that failed LOS model creation for no immediately obvious reason. Thus, support for tracking the SCA where GCPs were measured is retained in this algorithm.
5) A configuration table (system table) should be provided for each installation of the algorithm implementation to convey site-specific information such as the processing center name (used in the standard report header), the number of processors available (for parallel processing implementations), etc. This takes the place of the heritage system table, which also contained certain algorithm-related parameters. Anything related to the algorithms has been moved to the CPF for Landsat 8/9.

4.2.7 OLI Geodetic Accuracy Assessment (L1GT)

4.2.7.1 Background/Introduction

The OLI geodetic accuracy assessment, or geodetic characterization, algorithm analyzes the results of the GCP measurements created by the LOS model correction algorithm to assess the geolocation accuracy of the systematically terrain-corrected OLI L1G image used for GCP mensuration. Statistics are computed for the original (unadjusted) GCP measurements and for the final (best-fit) adjusted GCP locations. In both cases, GCPs identified as outliers by the LOS model correction algorithm are excluded. The “pre-fit” results, based on the unadjusted GCP measurements, provide a measure of L8/9 pointing, position, and alignment knowledge, as reflected in the measured geolocation accuracy. The “post-fit” results, based on the control point residuals after the precision LOS model corrections are applied, provides a measure of how well the precision correction process is working and an indication of the quality of the derived model corrections. This is particularly important in the L8/9 environment in which precision correction using GCPs will be attempted on all scenes, even those with cloud cover, and it will be necessary to identify the cases where the control point matching and precision-correction process has failed. The LOS model correction algorithm will perform these tests operationally, and the geodetic accuracy assessment algorithm will not be executed for scenes where the precision correction process is known to have failed. The geodetic accuracy assessment results will help identify cases where a substandard precision-correction solution has been accepted, thereby assisting in tuning the parameters used to detect precision correction failures.

Geodetic accuracy assessment will be performed as part of the processing flow for the standard L1TP scenes, processed using GCPs extracted from the Landsat Global Land Survey (GLS) data. It will also be performed during calibration site processing, using the more accurate GCPs derived from DOQ control. Although the geodetic accuracy assessment process is the same for both of these uses, the results will be trended

separately, as the GLS results will be used to assess the quality of the GLS global control, as well as the accuracy of the L8/9 products (see note #2), whereas the high-accuracy DOQ control will be used to assess the performance of the operational L8/9 navigation and geometric calibration (see note #1).

4.2.7.2 Dependencies

The OLI geodetic accuracy assessment algorithm assumes that the LOS model correction algorithm 4.2.3, and its predecessors, have been executed to create an output GCP residuals file. This file is parsed by the geodetic accuracy algorithm to extract the residuals for all non-outlier GCPs for both the first (unadjusted) and last (final) iterations of the LOS model correction process. Note that these residuals are recorded as along- and across-track offsets by the LOS model correction algorithm.

4.2.7.3 Inputs

The geodetic characterization algorithm uses the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs
ODL file
Input residual file name
Output geodetic report file name
Level 1GT mensuration image file name
L0R ID (for trending)
Work Order ID (for trending)
Trending flag (on/off)
Level 1GT Image File Contents (see note #5 and Resampling Algorithm for details)
WRS Path/Row (for trending) from the image metadata/DDR
Scene acquisition date (for trending) from the image metadata/DDR
Scene acquisition type (for trending) from the image metadata/DDR
Scene roll angle (for trending/report file) from the image metadata/DDR
Residual File Contents (see LOS Model Correction Algorithm for details (see 4.2.3))
GCP Latitude/longitude/height
GCP outlier/valid flag
Cross-track and along-track pre-fit residuals
Cross-track and along-track post-fit residuals
GCP source (GLS or DOQ) (new)

4.2.7.4 Outputs

Geodetic Accuracy Report (output file and trending) (see Table 4-20 below for additional details)
Processing Information
Processing Date and Time
Processing Center/Location
Processing Software Version
Processed L1GT Image File Name
Dataset Information

Geodetic Accuracy Report (output file and trending) (see Table 4-20 below for additional details)
Spacecraft and Instrument Source (Landsat 8/9 and OLI/OLI-2)
Work Order ID
WRS Path/Row
Roll Angle (new)
Acquisition Type (Earth, Lunar, Stellar) (will always be Earth)
L0Rp ID
Acquisition Date (new)
GCP Information
GCP Source (new)
Number of valid GCPs
Mean latitude and longitude of GCPs
Pre-Fit Statistics
Mean, RMSE, Standard Deviation, Correlation Coefficient
Post-Fit Statistics
Mean, RMSE, Standard Deviation, Correlation Coefficient

4.2.7.5 Options

Trending on or off.

4.2.7.6 Procedure

Geodetic characterization is performed on the precision residual file. See the LOS Model Correction Algorithm in Section 4.2.3 for further details on the correction process and its results. Geodetic characterization calculates statistics for the post- and pre-fit residuals of the precision-correction process. This process allows analysis of the accuracy of LOS model and its performance when processing image data using only the spacecraft ancillary data, as compared to applying ground control.

Since the LOS model correction algorithm detects and flags outlier GCPs, the work of the geodetic accuracy assessment algorithm is limited to reading and parsing the output residual file created by LOS mode correction, and computing summary statistics for the results of the first and last solution iterations. These results are written to a report file, along with standard header fields, some of which are extracted from the L1G image metadata. Figure 4-40 shows the architecture for the geodetic characterization algorithm.

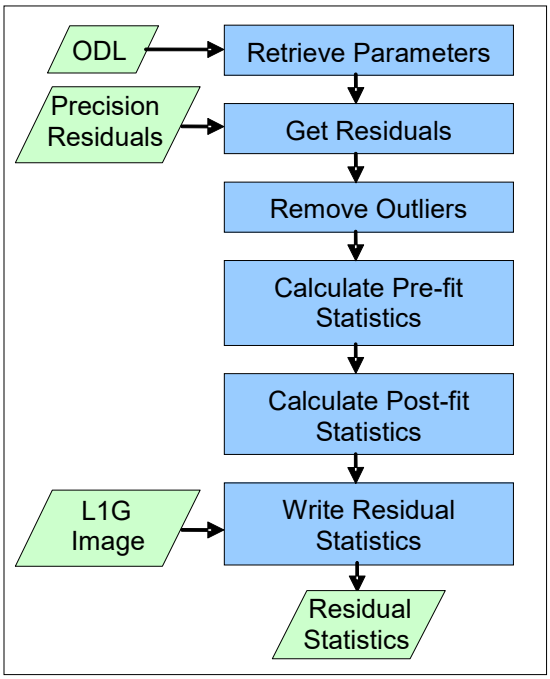

Figure 4-40. Geodetic Characterization Algorithm Architecture

4.2.7.6.1 Algorithm Flow

Inputs to the algorithm are an ODL parameter file, an ASCII residual file generated by the LOS correction algorithm, and the L1GT image used to measure the GCPs. Note that the L1GT image is used only to provide metadata for inclusion in the output report. The output is an ASCII report file containing a standard header that identifies the dataset analyzed, and the pre-fit and post-fit GCP residual summary statistics for the GCPs used in the precision correction solution.

Get Geodetic Characterization Parameters Sub-Algorithm

This sub-algorithm gets the parameters for geodetic characterization from the input ODL parameter file. It also reads the Level 1GT image metadata to retrieve the WRS path/row, acquisition date, acquisition type, and scene roll angle.

Read Grid Parameters Sub-Algorithm

This sub-algorithm reads the standard parameters common to all applications from the input ODL parameter file. These include the script name, Level 0R ID, work order ID, and trending on/off flag.

Get Residual Sub-Algorithm

This sub-algorithm reads the along- and across-track residual components for each GCP from the residual file created by the LOS model correction algorithm. It retrieves all of the residuals for a specified iteration. If a negative iteration number is provided, it retrieves the data for the final iteration. This sub-algorithm is invoked twice, once to retrieve the residuals for iteration 0 (pre-fit) and once to retrieve the final iteration residuals (post-fit). For each invocation, the entire input residual file is scanned until the

selected iteration header line is found (e.g., "Iteration 0" or "Final Iteration"). Then, each GCP record for that iteration is loaded into a residual data structure containing the following fields:

- Point Id
- Predicted L1GT Line
- Predicted L1GT Sample
- GCP Observation Time (seconds)
- GCP Latitude (degrees)
- GCP Longitude (degrees)
- GCP Height (meters)
- Across-Track View Angle (degrees)
- Across-Track Residual (meters)
- Along-Track Residual (meters)
- Image Y Residual (meters)
- Image X Residual (meters)
- Outlier/Valid Flag (0 = outlier, 1 = valid)
- GCP Source (DOQ or GLS)

Remove Outliers Sub-Algorithm
This sub-algorithm removes the residual records flagged as outliers from the GCP residual information and places the data in buffers to be accessed by later routines. Records with the outlier/valid flag field set to 0 are outliers.

Analyze GCP Residuals Sub-Algorithm
This sub-algorithm calculates the mean, RMSE, and standard deviation of the along- and across-track GCP residuals, as well as the correlation coefficient between the across- and along-track residuals. The statistics are computed in the following process:

a) Calculate GCP statistics
    a1) Calculate the total number of GCPs used (count of valid GCPs)
    a2) Calculate the mean latitude of GCPs used
    a3) Calculate the mean longitude of GCPs used

b) Calculate pre-fit statistics
    b1) Calculate the mean of cross-track residuals
    b2) Calculate the mean of along-track residuals
    b3) Calculate the RMSE of cross-track residuals
    b4) Calculate the RMSE of along-track residuals
    b5) Calculate the standard deviation of cross-track residuals
    b6) Calculate the standard deviation of along-track residuals
    b7) Calculate the correlation coefficient between along- and cross-track residuals

c) Calculate post-fit statistics
    c1) Calculate the mean of cross-track residuals

c2) Calculate the mean of along-track residuals
c3) Calculate the RMSE of cross-track residuals
c4) Calculate the RMSE of along-track residuals
c5) Calculate the standard deviation of cross-track residuals
c6) Calculate the standard deviation of along-track residuals
c7) Calculate the correlation coefficient between along- and cross-track residuals

The following equations are used to perform these calculations, with X being the parameter for which statistics are calculated:

Mean:

$$X_{mean} = \frac{1}{numGCP} \sum_{i=1}^{numGCP} X_i$$

RMSE:

$$X_{RMS} = \sqrt{\frac{1}{numGCP} \sum_{i=1}^{numGCP} X_i^{\,2}}$$

Standard Deviation:

$$X_{StdDev} = \sqrt{\frac{1}{numGCP - 1}\left(\left(\sum_{i=1}^{numGCP} X_i^{\,2}\right) - numGCP * X_{mean}^{\,2}\right)}$$

Correlation Coefficient:

$$XY_{CC} = \frac{\left(\sum_{i=1}^{numGCP} (X_i - X_{mean})(Y_i - Y_{mean})\right)}{(numGCP - 1)} \frac{1}{X_{StdDev}\, Y_{StdDev}}$$

Output Residual Statistics Sub-Algorithm
This sub-algorithm creates the output geodetic report file and writes the statistics computed from the GCP residuals to the output file. Note that the output of trending data to the characterization database is performed by the geodetic characterization main procedure.

Write Residual Statistics Sub-Algorithm
This sub-algorithm invokes output_header to write the standard report file header and then writes the GCP residual statistics to the ASCII output file.

Write Standard Report Header Sub-Algorithm
This sub-algorithm collects the input parameters, image metadata, and environment variable values needed to populate the standard IAS report header and writes the header to the ASCII output file.

4.2.7.6.2 Algorithm Output Details

Table 4-20 summarizes the geodetic accuracy assessment algorithm outputs. All fields are written to the output report file. Only those with "Yes" in the "Database Output" column are written to the characterization database. Note that the first eleven fields listed constitute the standard report header. Also note that the DEM Source field present in the heritage ALIAS implementation is no longer required.

Field	Description	Database Output
Date and time	Date (day of week, month, day of month, year) and time of file creation.	Yes
Spacecraft and instrument source	Landsat 8/9 and OLI/OLI-2	Yes
Processing Center	Processing center where the output was generated (see note #4).	Yes
Work order ID	Work order ID associated with processing (blank if not applicable).	Yes
WRS path	WRS path number	Yes
WRS row	WRS row number	Yes
Software version	Software version used to create report	Yes
Roll angle	Scene off-nadir roll angle (in degrees)	Yes
Acquisition Type	Earth viewing, Lunar, or Stellar (only Earth-viewing scenes are used for geodetic characterization)	Yes
L0Rp ID	Input L0Rp image ID	Yes
L1G image file	Name of L1GT used to measure GCPs	No
Acquisition date	Date of L1GT image acquisition	Yes
GCP source	Source of GCPs (GLS or DOQ)	Yes
Number of valid points	Number of GCPs accepted as valid	Yes
Mean GCP latitude	Mean latitude of valid GCPs (degrees)	Yes
Mean GCP longitude	Mean longitude of valid GCPs (degrees)	Yes
Pre-fit along-track mean	Mean of along-track iteration 0 residuals (meters)	Yes
Pre-fit across-track mean	Mean of across-track iteration 0 residuals (meters)	Yes
Pre-fit along-track RMSE	RMSE of along-track iteration 0 residuals (meters)	Yes
Pre-fit across-track RMSE	RMSE of across-track iteration 0 residuals (meters)	Yes
Pre-fit along-track standard deviation	Standard deviation of along-track iteration 0 residuals (meters)	Yes
Pre-fit across-track standard deviation	Standard deviation of across-track iteration 0 residuals (meters)	Yes
Pre-fit correlation coefficient	Correlation coefficient between along- and across-track iteration 0 residuals (dimensionless)	Yes
Post-fit along-track mean	Mean of along-track final iteration residuals (meters)	Yes
Post-fit across-track mean	Mean of across-track final iteration residuals (meters)	Yes
Post-fit along-track RMSE	RMSE of along-track final iteration residuals (meters)	Yes
Post-fit across-track RMSE	RMSE of across-track final iteration residuals (meters)	Yes

Field	Description	Database Output
Post-fit along-track standard deviation	Standard deviation of along-track final iteration residuals (meters)	Yes
Post-fit across-track standard deviation	Standard deviation of across-track final iteration residuals (meters)	Yes
Post-fit correlation coefficient	Correlation coefficient between along- and across-track final iteration residuals (dimensionless)	Yes

Table 4-20. Geodetic Accuracy Assessment Output Details

4.2.7.6.3 Accessing the Geodetic Accuracy Characterization Database

Although not part of the formal geodetic accuracy assessment algorithm, some comments regarding the anticipated methods of accessing and analyzing the geodetic accuracy results stored in the characterization database may assist with the design of the characterization database.

The database output from the geodetic accuracy assessment algorithm will be accessed by a statistical summary analysis tool that queries the characterization database to retrieve geodetic accuracy results from multiple scenes. Summary mean and RMSE statistics for the pre-fit scene results will be calculated and output in a report containing a comma-delimited table of the retrieved trending results, as well as the summary statistics.

The geodetic results would typically be queried by acquisition date, roll angle, WRS path/row, and/or GCP source. The most common query would be a combination of GCP source, roll angle, and acquisition date range, for example, selecting all of the GLS-derived results, from nadir scenes, for a given calendar quarter:

GCP_Source = "GLS"
Roll_Angle is between -0.5 and 0.5
Acquisition_Date is between 01APR2012 and 30JUN2012

Since we will be using the roll angle field to detect off-nadir acquisitions, it would be convenient if the associated query could be specified as a maximum (absolute) number (e.g., 0.5 degrees) rather than having to specify a plus/minus range.

The summary mean and RMSE statistics would be calculated from the pre-fit and post-fitmean, and RMSE results for the individual scenes returned as follows:

$$Mean_{net} = \frac{1}{numScene} \sum_{i=1}^{numScene} Mean_i$$

$$RMSE_{net} = \sqrt{\sum_{i=1}^{numScene} RMSE_i^{\,2} \Big/ numScene}$$

The query results would be formatted in a set of comma-delimited records (for ease of ingest into Microsoft Excel), one record per scene. Each record would contain all of the fields written to the characterization database (items with "Yes" in the rightmost column of Table 4-20 above). A header row containing the field names should precede the database records. Two trailer rows, one containing the summary statistic names (Net Pre-fit Along-Track Mean, Net Pre-fit Across-Track Mean, Net Pre-Fit Along-Track RMSE, Net Pre-Fit Across-Track RMSE, Net Post-Fit Along-Track Mean, Net Post-Fit Across-Track Mean, Net Post-Fit Along-Track RMSE, and Net Post-Fit Across-Track RMSE) and the second containing the comma-delimited summary statistic values, should follow the database records.

4.2.7.7 Notes

Some additional background assumptions and notes include the following:

- The pre-fit mean and RMSE statistics derived from DOQ control capture the absolute geodetic accuracy performance of the L8/9 system, whereas the standard deviation statistics reflect the relative geodetic accuracy.
- The post-fit RMSE statistics provide an indication of the absolute accuracy of the output L1T product, but this must be combined with an assessment of the accuracy of the GCP source to obtain a more realistic estimate of L1T accuracy. L1T accuracy is measured directly by the geometric accuracy assessment algorithm, which is a variant of the geodetic accuracy assessment process, but is documented as a separate algorithm (see Geometric Accuracy Assessment Algorithm in Section 4.2.6).
- The per-scene post-fit along-track RMSE and post-fit across-track RMSE statistics, derived from the GLS control used for L1TP product generation, would be good candidates for use as geometric quality metrics. The post-fit RMSE statistics could be extracted from either the geodetic accuracy assessment report file or the characterization database. In the case of a LOS model correction failure, fill values would be inserted into these quality fields to indicate that the registration to the GLS control failed, for example, due to cloud cover.
- A configuration table (system table) and/or environment variables should be provided for each installation of the algorithm implementation to convey site-specific information such as the processing center name (used in the standard report header), the number of processors available (for parallel processing implementations), etc. This takes the place of the heritage system table, which also contained certain algorithm-related parameters. Anything related to the algorithms has been moved to the CPF for Landsat 8/9.
- The input L1GT image is only used to extract selected metadata (noted in the input table) for inclusion in the output report and trending data. If the required fields are all available in the L0Rp dataset, it could be used as an input instead of the L1GT.

4.2.8 OLI Image Registration Accuracy Assessment Algorithm

4.2.8.1 Background/Introduction

The OLI Image Registration Accuracy Assessment, or Image-to-Image (I2I) characterization, algorithm has two purposes; it can be used to determine the geometric registration of an image to a particular source image, or it can be used to verify the multi-temporal capabilities of the OLI product generation system.

The I2I characterization process works by choosing locations within the reference and search images, extracting windows of imagery from each image, and performing gray scale correlation on the image windows. Several criteria are used in determining whether the correlation processing was successful. These criteria include measured displacement and strength of the correlation peak. The subpixel location of the measured offset is calculated by fitting a 2^{nd} order polynomial around the correlation surface and solving for the fractional peak location of the fitted polynomial. Once the total offset has been measured, adding the calculated integer offset to the calculated subpixel offset for all successfully correlated tie-points, a final t-distribution outlier rejection is performed to produce the set of all valid measured offsets.

Two options are available for determining tie-point locations in the I2I characterization algorithm. These options include choosing evenly spaced points in the output space of the imagery or choosing points based on pre-chosen locations.

4.2.8.2 Dependencies

The OLI I2I algorithm assumes a cloud-free L1TP has been generated and that a suitable reference image (OLI or other source) exists for comparison purposes. The L1TP needs to be in the SCA combined format.

4.2.8.3 Inputs

The Image Registration Accuracy Assessment algorithm and its component sub-algorithms use the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs	Source
Reference image	ODL
Search image	ODL
Bands to process	ODL
Tie point type	ODL
Number points in line direction (if tie-point type is evenly spaced)	ODL
Number points in sample direction (if tie-point type is evenly spaced)	ODL
GCP File Name (if tie-point type is file based)	ODL
Correlation window size lines	CPF/ODL
Correlation window size samples	CPF/ODL
Fill range maximum	CPF/ODL
Fill range minimum	CPF/ODL

Algorithm Inputs	Source
Fill percentage	CPF/ODL
T-distribution outlier threshold	ODL
Output file names	
I2I residuals file (see Table 4-22 below for details)	ODL
I2I output data file (see Table 4-21 below for details)	ODL
I2I statistics file (see Table 4-23 below for details)	ODL
Trend flag	ODL
L0R/L1R ID	
Work Order ID	
WRS Path/Row	
Trending thresholds (RMS for each line, sample per band – See note #5)	CPF
Minimum correlation peak	CPF/ODL
Maximum displacement	CPF/ODL
Correlation fit method (See note #2)	CPF

4.2.8.4 Outputs

I2I residuals file (See Table 4-22)
I2I data file (See Table 4-21)
I2I statistics file (See Table 4-23)
I2I characterization trending (if trending flag set to yes)
L0R/L1R ID
Work Order ID
WRS Path/Row
Reference source
I2I statistics (Min, Mean, Max, Median, RMS, Std. Dev.)

The processing parameters, listed in the table above and described in the following subsections, can be overridden if they are given as fields within the input ODL file: correlation window size, maximum offset, minimum correlation strength, fill threshold, maximum and minimum file values.

4.2.8.5 Options

Trending on/off switch
Correlation fit method (placeholder, see note #2)
Normalized gray scale or least squares correlation (see note #2)

4.2.8.6 Procedure

I2I characterization is used to assess the ability to register an OLI dataset to another image dataset. I2I characterization performs image correlation between OLI imagery, the search dataset, and a reference image dataset. Landsat, reduced resolution DOQ data, or OLI imagery can be used as a reference dataset. When OLI imagery is measured against another OLI dataset acquired at a different date over the same geographic area, I2I measures the ability to register multi-temporal OLI imagery. Correlation points are chosen either as evenly distributed points throughout the imagery or at predefined GCP locations. Outliers are first rejected by removing all measured

displacements that lie above a user set threshold or those whose correlation peak is below a given minimum value. A final outlier rejection is performed on the measured offsets using a Student-t distribution test. Final statistics, which are reported in the output statistics file and stored in the database, are calculated based on the valid displacements after the Student-t outlier rejection. Statistics are calculated for both the line and sample direction independently. Figure 4-41 shows an overview of the algorithm procedure.

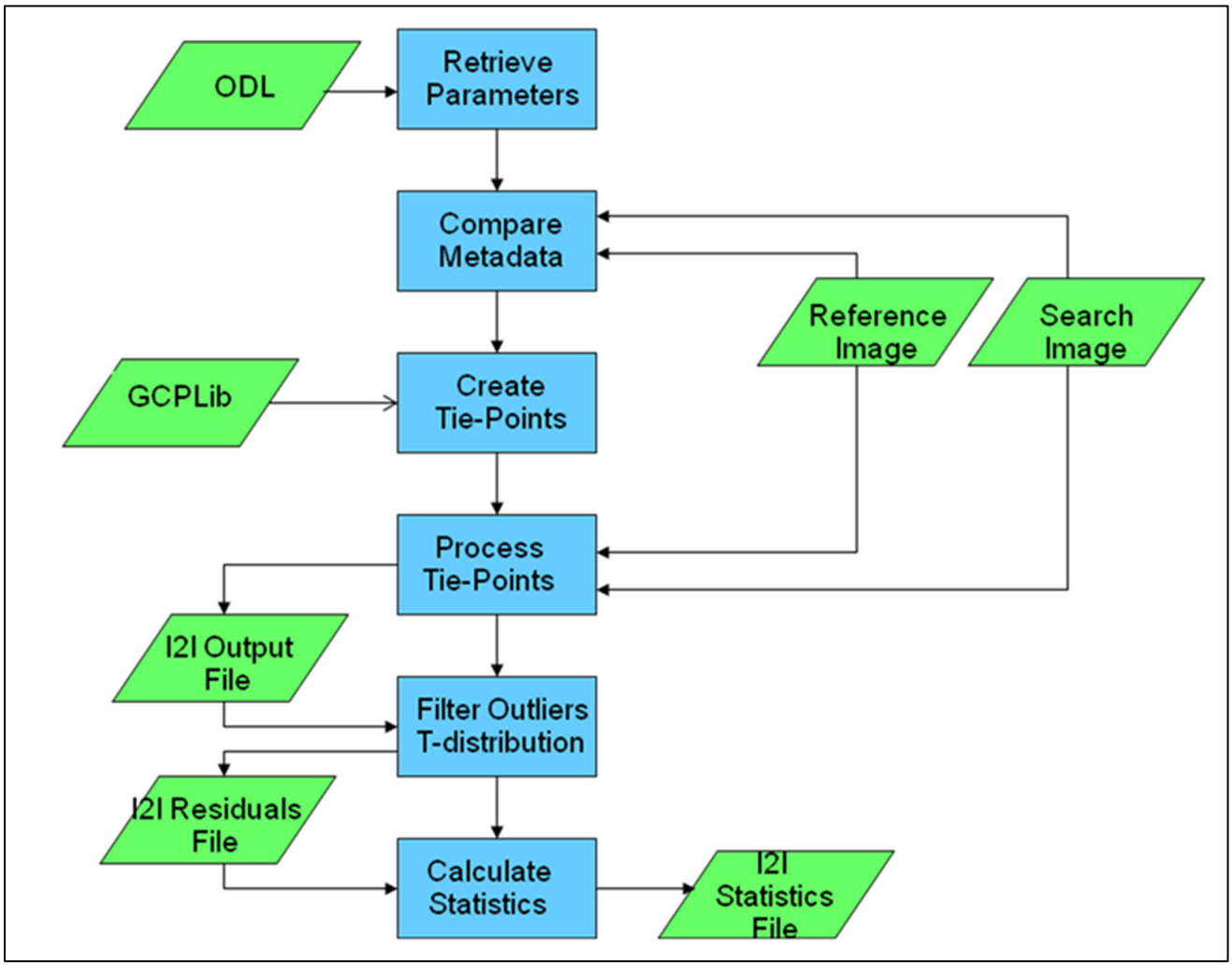

Figure 4-41. Image Registration Accuracy Assessment Algorithm Flow Diagram

The OLI Image Registration Accuracy Assessment algorithm contains a number of sub-algorithms that are shared with other algorithms. The correlation and mensuration modules, for example, are not described within this algorithm description, as they are already presented in the Ground Control Correlation (4.1.6) and Band Registration Accuracy Assessment (BRAA) (4.2.9) Algorithms. Those descriptions should be reviewed for any information pertaining to these processes. Explanations of the methodology of the mensuration and t-distribution outlier rejection processes are in the

Band Registration Accuracy Assessment Algorithm. Please review that description for any information pertaining to these methodologies.

4.2.8.6.1 Stage 1 - Data Input

The data input stage involves loading the information required to perform the image registration assessment. This includes reading the image files, retrieving the output I2I file names: data, residuals and statistic files; retrieving or initializing processing parameters: maximum displacement, fill range, fill threshold, minimum correlation peak, t-distribution threshold, bands to process, correlation window size, trending thresholds, tie-point method; and if the tie-point method is set to file-based, the GCP file name. Once the input file (and, if needed, the GCP name) has been retrieved, the files and the information stored within them can be opened and read.

4.2.8.6.2 Stage 2 - Create Tie-point Locations

Tie point locations may be determined in an evenly spaced pattern in output space, or they may be read from a GCP file.

Determine Evenly Spaced Tie-points

This tie-point selection process is similar to the Band Registration Accuracy Assessment Algorithm, Stage 3 - Create Tie-point Locations, section Determine Evenly Spaced Tie-points. The difference between the two processes is that the Image Registration Accuracy Assessment algorithm computes and uses the bounding area between the search and references for tie-point selection.

Creating Evenly Space Tie-Points Processing Steps

1. Map the search corners to the reference space.

$$\text{search line}_i = \frac{Y_{ULref} - Y_i}{P_y} + 1$$

$$\text{search sample}_i = \frac{X_i - X_{ULref}}{P_x} + 1$$

Where
i = 0,1,2,3 for the search upper-left, upper-right, lower-right, lower-left coordinates
Y_{ULref} = Reference the upper-left Y coordinate
X_{ULref} = Reference the upper-left X coordinate
P_x = pixel size sample direction
P_y = pixel size line direction

2. Determine the bounding overlapping area.
minimum sample = min(search sample$_i$, reference sample$_i$)
maximum sample = max(search sample$_i$, reference sample$_i$)
minimum line = min(search line$_i$, reference line$_i$)
maximum line = max(search line$_i$, reference line$_i$)

3. Calculate the step sizes.

$$\text{spacing x} = \frac{\text{maximum sample} - \text{minimum sample} - \text{correlation window samples}}{\text{M} - 1}$$

$$\text{spacing y} = \frac{\text{maximum line} - \text{minimum line} - \text{correlation window lines}}{\text{N} - 1}$$

Where
M = Number of tie-points in the sample direction
N = Number of tie-points in the line direction

4. Set the evenly spaced tie-point locations.
   4.1 For j = 0 to N-2

$$\text{tie-point location y}[j] = \frac{\text{correlation window lines}}{2} + j * \text{spacing y}$$

   4.2 $\text{tie-point location y}[N-1] = \text{maximum line} - \frac{\text{correlation window lines}}{2}$

   4.3 For i = 0 to M-2

$$\text{tie -point location x}[i] = \frac{\text{correlation window samples}}{2} + i * \text{spacing x}$$

   4.4 $\text{tie -point location x[M-1]} = \text{maximum sample} - \frac{\text{correlation window samples}}{2}$

Determine File-Based Tie-points

This tie-point selection is based on an input ASCII file containing latitude and longitude locations for each individual tie-point. These individual locations are converted to line and sample locations within the search and reference images. These locations then have windows of imagery extracted from the search and reference images, after which displacement between the two windowed images can be calculated. This file is called the GCPLib, and is referred to as the ASCII GCP file within the text of this document.

Create Tie-Point from GCP ASCII (GCPLib) file Processing Steps

1. Open the GCP ASCII (GCPLib) file.
2. For each GCP:
   2.1 read GCP (note #1).
   Chip ID and name
   Latitude and longitude
   Projection X, Y, Z
   2.2 Convert the GCP geographic/projection location to the line and sample locations within the image files.
      2.2.1 Convert the GCP geographic location to the search/reference map projection. Map projection conversions can be done through the General Cartographic Transformation Package (GCTP).
      2.2.2 Convert the map projection X and Y locations to the line and sample locations.
         Equations:

$$line = \frac{Y_{GCP} - Y_{UL}}{P_y} + 1$$

$$sample = \frac{X_{UL} - X_{GCP}}{P_x} + 1$$

Where
Y_{UL} = Upper-left Y coordinate of the image
X_{UL} = Upper-left X coordinate of the image
Y_{GCP} = Y coordinate of GCP
X_{GCP} = X coordinate of GCP
P_x = Image pixel size in the sample direction
P_y = Image pixel in size in the line direction

2.2.2.a Convert to the line and sample location in the search image.
2.2.2.b Convert to the line and sample location in the reference image.

2.3 Store the line and sample locations for search and reference.

3. Close the GCP ASCII (GCPLib) file.

4.2.8.6.3 Stage 3 - Calculate Individual Point-by-Point Image Displacements

Normalized cross-correlation is used to measure spatial differences between the reference and search windows extracted from the imagery to be compared. The normalized cross-correlation process helps to reduce any correlation artifacts that may arise from radiometric differences between the two image sources. The correlation process will only measure linear distortions over the windowed areas. By choosing appropriate correlation windows that are well distributed throughout the imagery, non-linear differences between the image sources can be found. The Band Registration Accuracy Assessment Algorithm explains this methodology.

4.2.8.6.4 Stage 4 - Removing Outliers Using the t-distribution

Once all of the line and sample offsets have been measured and the first level of outlier rejection has been performed, and a check against the maximum allowable offset and the minimum allowable correlation peak takes place, the measurements are further filtered for outliers using a Student-t outlier rejection. The Band Registration Accuracy Assessment Algorithm explains this methodology. Please reference that description for these items.

Image Accuracy Assessment Processing Steps

1. For band = Number of OLI bands to process
   1.1. For index = Number of tie-points to process
      1.1.1. Read the current tie-point chip and tie-point location.
             Set the tie-point flag to unsuccessful.
      1.1.2. Extract the search window (of imagery) at the tie-point location.
      1.1.3. Extract the reference window (of imagery) at the tie-point location.

1.1.4. Count the number of pixels in the reference window that is within the fill range.

count = 0

For i=0 to the number of pixels in the correlation window

If the reference pixel is > fill min and the reference pixel is < fill max

count++

1.1.5. Check the number of reference pixels counted against the fill threshold/percentage.

$$if: \frac{count}{correlation\ window\ size} > fill\ threshold$$

increment the index to the next tie-point location

$$else: continue$$

1.1.6. Count the number of pixels in the search window that is within the fill range.

count = 0

For i=0 to number of pixels in correlation window

If the search pixel is > fill min and search pixel is < fill max

count++

1.1.7. Check the number of search pixels counted against the fill threshold/percentage.

$$if: \frac{count}{correlation\ window\ size} > fill\ threshold$$

increment index to next tie-point location

$$else: continue$$

1.1.8. Perform normalized gray scaled correlation between the reference and search windowed images, calculating correlation surface *R* (See Band Registration Accuracy Assessment Algorithm - Stage 4 Calculate Individual Point-by-Point Band Displacements).

1.1.9. Find the peak within the correlation surface.

Max = *R*(0,0)

For i=0 to correlation window number of lines -1

For j=0 to correlation window number of samples -1

If *R*(i,j) > max then

Max = *R*(i,j)

line offset = i

sample offset = j

1.1.10. Check the correlation peak against the threshold.

if max > peak threshold

continue

else

set tie-point flag to outlier and choose the next tie-point

1.1.11 Measure the subpixel peak location (See Band Registration Accuracy Assessment Algorithm - Stage 4 Calculate Individual Point-by-Point Band Displacements)
Δsub-line
Δsub-sample
1.1.12. Calculate the total pixel offset.
total line offset = line offset + Δsub-line
total sample offset = sample offset + Δsub-sample
1.1.13. Check the offset against the maximum displacement offset.

$$\text{total displacement} = \sqrt{(\text{total line offset})^2 + (\text{total sample offset})^2}$$

if (total displacement > maximum displacement)
Set the tie-point flag to outlier and choose the next tie-point
else
Set the tie-point flag to valid
1.2 Store the band tie-point mensuration information, correlation success, and offsets measured. See Table 4-21.
2. For band = 1 to Number of bands to process
2.1 Perform t-distribution outlier rejection (See Band Registration Accuracy Assessment - Stage 5 Removing Outliers Using the t-distribution).
2.2. Store the band combination final individual tie-point information and outlier flag. See Table 4-22.
3. For band combination = 1 to Number of band combinations
3.1. Calculate mean, minimum, maximum, median, standard deviation, and root mean squared offset.
3.2. Store the band combination statistics. See Table 4-23.
4. Perform trending if trending flag is set to yes.
4.1. Check the results against the trending thresholds.
For each band
if measured RMSE > trending thresholds
exit trending
If there are no RMSE > trending thresholds perform trending

4.2.8.6.5 Output files

The output files are listed below for the Image Registration Accuracy.

All output files contain a standard header. This standard header is at the beginning of the file and contains the following:

- Date and time the file was created
- Spacecraft and instrument pertaining to measurements
- Off nadir (roll) angle of spacecraft/instrument
- Acquisition type
- Report type (Image-to-Image)
- Work order ID of the process (left blank if not applicable)
- WRS path/row
- Software version that produced the report
- L0R image file name

The data shown in Table 4-21, Table 4-22, and Table 4-23 are stored in the database. The statistics stored per band will be used for trending analysis of the image registration accuracy of the OLI instrument. Results produced through a time-series analysis of these data stored, over a set time interval or multiple image files, will be used for a temporal assessment of the registration quality of the OLI products. The SCA number fields are listed in the tables for Image Registration Accuracy Assessment for consistency with the tables listed in the Band Accuracy Assessment Algorithm.

1.	Field	2.	Description
3.	Date and time	4.	Date (day of week, month, day of month, year) and time of file creation.
5.	Spacecraft and instrument source	6.	Landsat 8/9 and OLI/OLI-2 (TIRS/TIRS-2 if applicable)
7.	Processing Center	8.	EROS Data Center SVT
9.	Work order ID	10.	Work order ID associated with processing (blank if not applicable)
11.	WRS path/row	12.	WRS path and row
13.	Software version	14.	Software version used to create report
15.	Off-nadir angle	16.	Off-nadir roll angle of processed image file
17.	Acquisition Type	18.	Earth viewing or Lunar
19.	L0R image file	20.	L0R image file name used to create L1TP
21.	Processed image file name	22.	Name of L1TP used to create report
23.	Reference bands	24.	Reference bands used in image assessment
25.	Search bands	26.	Search bands used in image assessment
27.	Heading for individual tie-points	28.	One line of ASCII text defining individual tie-point fields.
29.	For each tie-point:	30.	
31.	Tie point number	32.	Tie-point index/number in total tie-point list
33.	Reference line	34.	Tie-point line location in reference image (band)
35.	Reference sample	36.	Tie-point sample location in reference image (band)
37.	Reference latitude	38.	Tie-point latitude location
39.	Reference longitude	40.	Tie-point longitude location
41.	Reference elevation	42.	Elevation of tie-point location
43.	Search line	44.	Tie-point line location in search image
45.	Search sample	46.	Tie-point sample location in search image
47.	Delta line	48.	Measured offset in line direction
49.	Delta sample	50.	Measured offset in sample direction
51.	Outlier flag	52.	1=Valid, 0=Outlier
53.	Reference band	54.	Reference band number
55.	Search band	56.	Search band number
57.	Reference SCA	58.	SCA number that reference window was extracted from
59.	Search SCA	60.	SCA number that search window was extracted from
61.	Search image	62.	Name of search image
63.	Reference image	64.	Name of reference image

Table 4-21. Image Registration Accuracy Assessment Data File

65. Field	66. Description
67. Date and time	68. Date (day of week, month, day of month, year) and time of file creation.
69. Spacecraft and instrument source	70. Landsat 8/9 and OLI/OLI-2 (TIRS/TIRS-2 if applicable)
71. Processing Center	72. EROS Data Center SVT
73. Work order ID	74. Work order ID associated with processing (blank if not applicable)
75. WRS path/row	76. WRS path and row
77. Software version	78. Software version used to create report
79. Off-nadir angle	80. Off-nadir pointing angle of processed image file
81. Acquisition Type	82. Earth viewing or Lunar
83. L0R image file	84. L0R image file name used to create L1TP
85. Processed image file name	86. Name of L1TP used to create report
87. Number of records	88. Total number of tie-points stored in file
89. Heading for individual tie-points	90. One line of ASCII text defining individual tie-point fields.
91. For each band combination	92.
93. Combination header	94. Number of points in combination, reference band number, search band number.
95. For each tie-point:	96.
97. Tie point number	98. Tie-point index/number in total tie-point list
99. Reference line	100. Tie-point line location in reference image (band)
101. Reference sample	102. Tie-point sample location in reference image (band)
103. Reference latitude	104. Tie-point latitude location
105. Reference longitude	106. Tie-point longitude location
107. Reference elevation	108. Elevation of tie-point location
109. Search line	110. Tie-point line location in search image
111. Search sample	112. Tie-point sample location in search image
113. Delta line	114. Measured offset in line direction
115. Delta sample	116. Measured offset in sample direction
117. Outlier flag	118. 1=Valid, 0=Outlier
119. Reference band	120. Reference band number
121. Search band	122. Search band number
123. Reference SCA	124. SCA number that reference window was extracted from
125. Search SCA	126. SCA number that search window was extracted from
127. Search image	128. Name of search image
129. Reference image	130. Name of reference image

Table 4-22. Image Registration Accuracy Assessment Residuals File

131. Field	132. Description
133. Date and time	134. Date (day of week, month, day of month, year) and time of file creation.
135. Spacecraft and instrument source	136. Landsat 8/9 and OLI/OLI-2 (TIRS/TIRS-2 if applicable)
137. Processing Center	138. EROS Data Center SVT
139. Work order ID	140. Work order ID associated with processing (blank if not applicable)
141. WRS path/row	142. WRS path and row
143. Software version	144. Software version used to create report
145. Off-nadir angle	146. Off-nadir pointing angle of processed image file
147. Acquisition Type	148. Earth viewing or Lunar
149. L0R image file	150. L0R image file name used to create L1TP
151. Processed image file name	152. Name of L1TP used to create report
153. t-distribution threshold	154. Threshold used in t-distribution outlier rejection
155. For each band	156.
157. Reference band	158. Reference band of statistics
159. Search band	160. Search band of statistics
161. SCA	162. SCA number of search image
163. Total tie-points	164. Total number of tie-points for band
165. Correlated tie-points	166. Number of tie-points that successfully correlated for band
167. Valid tie-points	168. Total number of valid tie-points for band after all outlier rejection has been performed
169. For both line and sample direction:	170. All statistics are given in terms of pixels
171. Minimum offset	172. Minimum offset within all valid offsets
173. Mean offset	174. Mean offset of all valid offsets
175. Maximum offset	176. Maximum offset within all valid offsets
177. Median offset	178. Median offset within all valid offsets
179. Standard deviation	180. Standard deviation of all valid offsets
181. Root-mean-squared	182. Root mean squared offset of all valid offsets

Table 4-23. Image Registration Accuracy Assessment Statistics Output File

4.2.8.7 Notes

Some additional background assumptions and notes include the following:

1) The GCP structure and retrieval modules are set up to be generic. This structure contains the following for each GCP:
    - Point ID
    - Chip name
    - Reference line and sample
    - Latitude, Longitude
    - Projection X, Y, Z
    - Pixel size Y, X
    - Image chip size line, sample

    - Source of GCP
    - Date of GCP
    - Relative/absolute flag
2) This type of generic GCP structure and management will work for OLI processing also.
3) The correlation result fit method defines the algorithm used to estimate the correlation peak location to subpixel accuracy. Only the quadratic surface fitting method described in this algorithm description is supported in the baseline algorithm. Note that the fine least-squares correlation method, invoked by selecting correlation windows with an odd number of lines or samples, does not use a separate peak finding method.
4) Image Registration Accuracy statistics stored within the database will be accessed for analysis.
    - *Accessed according to a specific date range.*
    - *Accessed according to a specific band.*
    - *Accessed according to a specific geographic location.*
    - *Accessed according to acquisition type (nadir, off-nadir, lunar).*
5) These data accessed can be retrieved and stored within a comma-delimited file. The methodology used to access the database could be an SQL script.
6) Data stored within the database will be accessed for time series analysis.
    - *Data would be pulled for a user-specified time.*
    - *Statistics over multiple scenes would be calculated and combined into band- or scene-based statistics.*
7) These calculations could be performed within the methodology used to access the data from the database (SQL script).
8) There will need to be a set of criteria, based on calculated statistics, as to whether trending should be performed. These criteria would be provided to avoid having garbage stored in the database. Any values needed in determining whether the criteria have been met for trending would be stored and retrieved from the CPF. There would be one threshold per band. Step 4.1 of the Image Accuracy Registration Assessment Processing steps section provides the criteria to check for trending.

4.2.9 OLI Band Registration Accuracy Algorithm

4.2.9.1 Background/Introduction

The OLI Band Registration Accuracy Assessment (BRAA) Algorithm, or the Band-to-Band (B2B) Characterization process, measures the relative band alignment between all bands of each Sensor Chip Assembly (SCA) for the OLI instrument. The displacement for every pair-wise combination of all bands of each SCA requested for assessment is measured; creating an over-determined dataset of band-to-band measurements for each SCA. The residuals measured from the B2B characterization process will be used to assess the accuracy of the band-to-band registration of the OLI instrument, and if needed, used as input to the band calibration algorithm in order to calculate new LOS parameters for the CPF.

The B2B characterization process works by choosing tie-point locations within band pairs of each SCA, extracting windows of imagery from each band, and performing gray scale correlation on the image windows. Several criteria are used in determining whether the correlation processing was successful. These criteria include measured displacement and strength of the correlation peak. The subpixel location of the measured offset is calculated by fitting a 2nd order polynomial around the discrete correlation surface and solving for the fractional peak location of the fitted polynomial. The total offset measured is then the integer location of the correlation peak plus the subpixel location calculated. A new fine-resolution least-squares correlation method has been added to the heritage algorithm to provide a more accurate measurement of subpixel offsets. This method is described below.

Several options are available for processing data through the BRAA algorithm. These include choosing evenly spaced points for the location of the windows extracted, choosing to use the geometric grid for determining window locations in order to avoid fill within the image files, specifying the bands and/or the SCA to process, and specifying the valid pixel range to use during correlation. The least-squares correlation method is invoked by requesting image windows with at least one odd dimension, since the heritage algorithm only works with images with even dimensions (e.g., 32x32 image windows will use normalized gray scale correlation, but 31x31 image windows will use least-squares correlation).

With the exception of a few high-altitude calibration sites where the ground surface is visible in the cirrus band, Earth-based acquisitions will be used to characterize all bands except the cirrus. Lunar acquisitions will also be used to characterize the cirrus band. Both acquisition types will be passed through BRAA. In terms of the BRAA, it does not matter which acquisition type is being passed into the algorithm; some of the processing parameters and options may change due to the acquisition type, but both types will use the same mensuration process to create an assessment of the band registration.

4.2.9.2 Dependencies

The OLI BRAA assumes that a cloud-free Earth-viewing L1TP or Lunar L1G image has been generated, and depending on the tie-point selection type chosen, that the LOS Model Correction (6.2.3) and the LOS Projection and Gridding (6.2.2) algorithms have been executed to create a geometric grid file. The L1TP/L1G image needs to be in the SCA-separated format and either in a SOM or UTM path-oriented projection for Earth acquisitions. For best results, the DOQ control and best available DEM should be used in generating the L1TP.

4.2.9.3 Inputs

The BRAA and its component sub-algorithms use the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs	Source
ODL file (implementation)	
Calibration Parameter File (baseline)	ODL

Algorithm Inputs	Source
Correlation Fit Method (see note #14)	CPF
Correlation Window Size	CPF or ODL
Correlation Maximum Displacement	CPF or ODL
Correlation Fill Threshold	CPF or ODL
Correlation Minimum Fill Value	CPF or ODL
Correlation Maximum Fill Value	CPF or ODL
L1TP/L1G image	ODL
OLI resampling grid (optional)	ODL
Outlier (t-distribution) threshold	ODL
B2B characterization output file	ODL
Output residuals file name	ODL
Output statistics file name	ODL
SCAs to process	ODL
Bands to process	ODL
Fill range maximum	ODL
Fill range minimum	ODL
Fill threshold or percentage	ODL
Correlation window size lines	ODL
Correlation window size samples	ODL
Tie-point spacing in line direction	ODL
Tie-point spacing in sample direction	ODL
Trending flag	ODL
L0R ID (for trending)	ODL
Work Order ID (for trending)	ODL
Calibration Parameter File (baseline, if trending is requested)	ODL
Trending thresholds (Standard deviation line, sample per band per SCA - see note #3).	CPF

4.2.9.4 Outputs

Pan downsampled image
B2B residuals file (see Table 4-25 below for details)
B2B output data file (see Table 4-24 below for details)
B2B statistics file (see Table 4-26 below for details)
B2B characterization trending (if trend flag set to yes)
L0R/L1R ID
Work Order ID
WRS Path/Row
B2B statistics for all band combinations and SCAs

The processing parameters, listed in the table above and described in the following subsections, can be overridden if they are given as fields within the input ODL file: correlation window size, maximum offset, minimum correlation strength, fill threshold, maximum and minimum file values.

4.2.9.5 Options

Trending on/off switch
Grid-based tie-point generation
Normalized gray scale or least-squares correlation

4.2.9.6 Procedure

Band Registration Accuracy Assessment measures the misalignment between spectral bands after all known geometric effects have been taken into account. The results from the band registration assessment are used by the band alignment calibration routine (See Band-to-Band Calibration Algorithm in Section 4.2.10) to estimate new Legendre LOSs (See Line-of-Sight Model Creation Algorithm in Section 4.2.3) for each band of each SCA. Due to the different viewing angles for each band of each SCA, geometric displacement due to relief must be removed from the imagery for band-to-band characterization of Earth acquisitions, i.e., input imagery for band registration assessment must be precision- and terrain-corrected (see the Resampling Algorithm in Section 4.2.4). Figure 4-42 shows the steps involved in band registration assessment, and include creating datasets with common pixel resolutions; choosing locations (tie-point locations) for measurement; performing mensuration; removing outliers from calculated residuals; and calculating statistics from the remaining residuals. Residuals are measured for each band combination of each SCA that is requested through the input parameters.

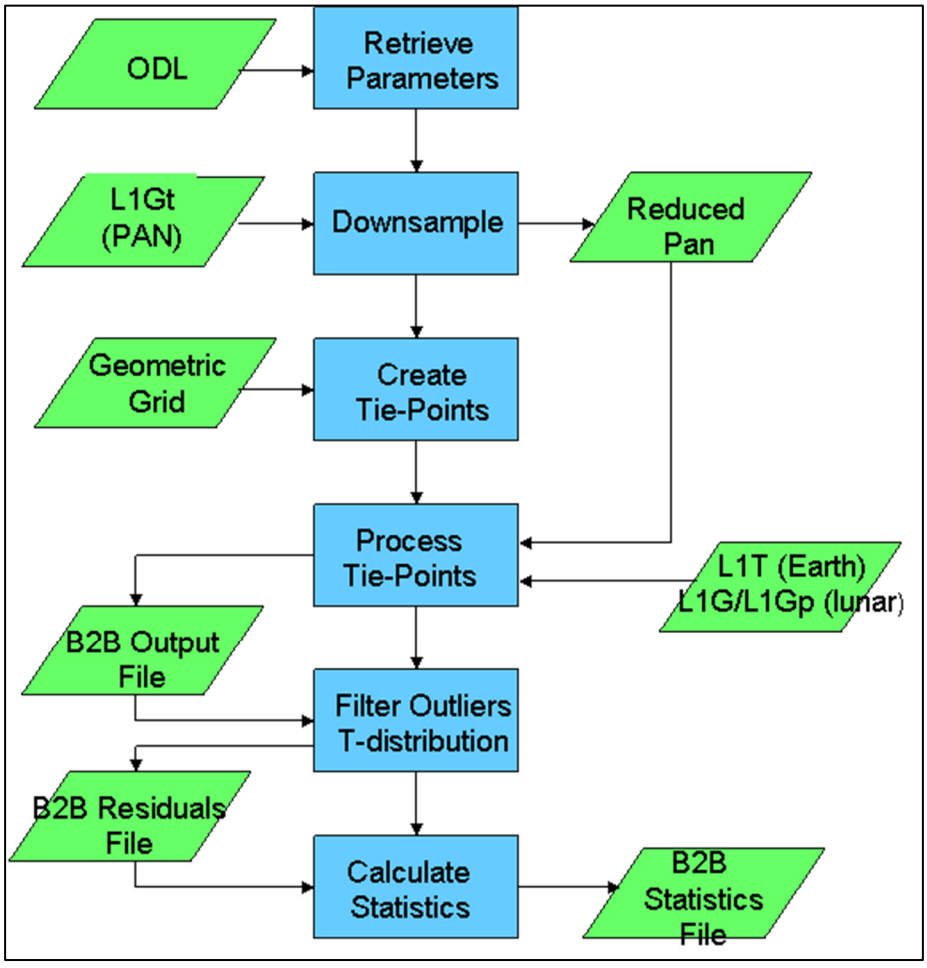

Figure 4-42. Band Registration Accuracy Block Diagram

The OLI Band Registration Accuracy Algorithm uses a number of modules that are shared with other algorithms. The correlation and mensuration modules, for example, are not described here, as they are already present within the Ground Control Point Correlation Algorithm in Section 4.1.6.

4.2.9.6.1 Stage 1 - Data Input

The data input stage involves loading the information required to perform the band registration assessment. This includes reading the image file, retrieving the output B2B file names: output, residuals and statistic files; retrieving or initializing processing parameters: maximum displacement, fill range, fill threshold, minimum correlation peak, t-distribution threshold, SCAs to process, bands to process, correlation window size, trending thresholds, tie-point method; and if the tie-point method is set to grid-based, the geometric grid file name will be read. Once the input file (and, if needed, the geometric grid name) has been retrieved, the files and the information stored within them can be opened and read.

4.2.9.6.2 Stage 2 - Creating a Reduced Resolution PAN band

Before displacement between the PAN band and the other multispectral bands can be measured, the PAN band must be reduced in resolution to match that of the multispectral bands. An oversampled cubic convolution function is used to reduce the resolution of the PAN band. Cubic convolution interpolation uses a set of piecewise cubic spline interpolating polynomials. The polynomials have the following form:

$$f(x) = \begin{cases} (\alpha+2)|x|^3 - (\alpha+3)|x|^2 + 1 & 0 \le |x| < 1 \\ \alpha|x|^3 - 5\alpha|x|^2 + 8\alpha|x| - 4\alpha & 1 \le |x| < 2 \\ 0 & |x| > 2 \end{cases}$$

Since the cubic convolution function is a separable function, a two-dimensional representation of the function is given by multiplying two one-dimension cubic convolution functions, one function representing the x-direction the other function representing the y-direction. For an offset of zero, or x = 0, and α = -1.0, the discrete cubic function has the following values: f(0) = 1 and f(n) = 0 elsewhere. Therefore, convolving the cubic convolution function of x = 0 with a dataset leaves the dataset unchanged.

$$y[n] = f[n] \otimes x[n]$$
$$\text{for } x = 0$$
$$\text{gives } y[n] = x[n]$$
$$\text{where } \otimes \text{ is the convolution operator}$$

Figure 4-43 shows the cubic function f(t) (dashed line) and the corresponding discrete weights for an offset, or phase, of zero (crossed-dots).

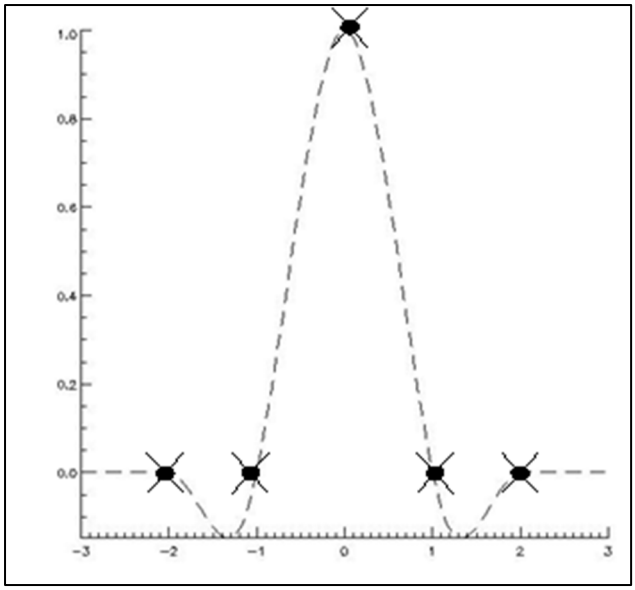

Figure 4-43. Cubic Convolution Function and Weights for Phase of Zero

To spatially scale an input data stream, an oversampled cubic convolution function with an offset of x =0 can be used. This can best be understood by looking at the Fourier Transform scaling property of a function that is convolved with a given input dataset:

$$f(t) \otimes x(t) \Leftrightarrow F(\omega) \bullet X(\omega)$$

$$x(at) = \frac{1}{|a|} X\left(\frac{\omega}{a}\right)$$

Where:

$\otimes$ is convolution
● is multiplication
F is the Fourier transform of *f*
X is the Fourier transform of *x*
t is time
ω is frequency

Applying the cubic function and scaling properties to an image data file shows that densifying the points applied with the cubic convolution function will, in turn, inversely scale the function in the frequency domain, therefore reducing the resolution of the imagery. By setting the cubic convolution offset to zero, densifying the number of weights of the cubic function, and convolving these weights to an image file, a reduction in resolution will be the resultant output image file. Figure 4-44 shows the cubic function with corresponding weights densified by a factor of two and a phase shift of zero. To ensure that the cubic weights do not scale the DNs of the output imagery during convolution, the cubic weights are divided by the scale factor.

$$f_s[n] = \frac{1}{2}\sum_{n=-4}^{4} f\left(\frac{n}{2}\right)$$

Where:

f_s[n] = scaled cubic convolution weights

f(n) = cubic convolution function

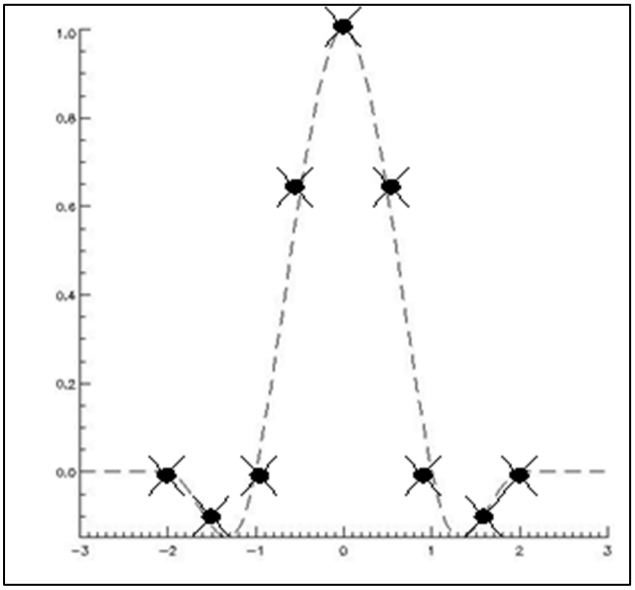

Figure 4-44. Cubic Convolution Densified by a Factor of 2

Scaling the cubic convolution function by a factor of 2 gives the following one-dimensional set of weights:

$$ccw[n] = [0.0 \quad -0.0625 \quad 0.0 \quad 0.3125 \quad 0.5 \quad 0.3125 \quad 0.0 \quad -0.0625 \quad 0.0]$$

To determine the two-dimensional cubic convolution weights, two one-dimensional sets of cubic weights are multiplied together (note only 7 values are needed for ccw, outside of this extent the weights are zero):

$$ccw[n] \times ccw[m] = ccw[n,m] = \begin{bmatrix} 0.0039 & 0.0 & -0.0195 & -0.0313 & -0.0195 & 0.0 & 0.0039 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.0195 & 0.0 & 0.0977 & 0.1563 & 0.0977 & 0.0 & -0.0195 \\ -0.0313 & 0.0 & 0.1563 & 0.25 & 0.1563 & 0.0 & -0.0313 \\ -0.0195 & 0.0 & 0.0977 & 0.1563 & 0.0977 & 0.0 & -0.0195 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0039 & 0.0 & -0.0195 & -0.0313 & -0.0195 & 0.0 & 0.0039 \end{bmatrix}$$

Where:

ccw[n] is a 8x1 one-dimensional set of cubic weights
ccw[m] is a 1x8 one-dimensional set of cubic weights

4.2.9.6.2.1 Procedure for Reducing PAN band

To reduce the resolution of the PAN band, apply the ccw[n,m] weights to the PAN image data:

reduced pan = ccv[n,m] * pan band

Note: The number of lines and number of samples listed below pertain to the size of the PAN band imagery.

Reduce PAN Band Resolution Processing Steps

1. Set line =0 then for every other PAN line
1.1. Set sample = 0 then for every other PAN sample
1.2. initialize summing variable sum = 0.0
1.3. For m = -4 to 4
    1.3.1. For n = -4 to 4
    1.3.2. Check to see if current image index is within valid imagery
    1.3.3. if m + line < 0 then line index = -m - line
        else if m + line >= number of lines then line index =
            2 * number of lines - m - line - 1
        else line index = m + line
    1.3.4. if n + sample < 0 then sample index = -n - sample
        else if n + sample >= number of sample then sample index =
            2 * number of samples - n - sample - 1
        else sample index = n + sample
    1.3.5. sum = sum + ccw[n+4,m+4] • pan[line index, sample index]
1.4. Store output DN for reduced PAN
    output line = line / 2
    output sample = sample / 2
    reduce pan[output line, output sample] = sum

4.2.9.6.3 Stage 3 - Create Tie-point Locations

Tie-point locations may be determined in an evenly spaced pattern in output space, or they may be established in an evenly spaced pattern in input space, using the OLI geometric grid.

4.2.9.6.3.1 Determine Evenly Spaced Tie-points (See notes #6 and #7)

To determine evenly spaced tie-point locations, a tie-point location is defined by stepping through the output space of the imagery by the user-defined steps N,M.

Create Evenly Spaced Tie-Points Processing Steps

1. Determine the number of tie-points in the sample and line direction:

$$tie-point\ spacing\ x = \frac{ONS-\ correlation\ window\ samples}{M-1}$$

$$tie-point\ spacing\ y = \frac{ONL-\ correlation\ window\ lines}{N-1}$$

Where:

M = user-entered number of tie-points in the sample direction
N = user-entered number of tie-points in the line direction
ONS = number of samples in the output space of the multispectral band
ONL = number of lines in the output space of the multispectral band
Correlation window samples = user-entered correlation window size in samples
Correlation window lines = user-entered correlation window size in lines

2. Set evenly spaced tie-point locations.

2.1. For j = 0 to N-2

$$tie-point\ location\ y[j] = \frac{correlation\ window\ lines}{2} + j * tie-point\ spacing\ y$$

2.2.

$$tie-point\ location\ y[N-1] = ONL - \frac{correlation\ window\ lines}{2}$$

2.3. For i = 0 to M-2

$$tie-point\ location\ x[i] = \frac{correlation\ window\ samples}{2} + i * tie-point\ spacing\ x$$

2.4.

$$tie-point\ location\ x[M-1] = ONS - \frac{correlation\ window\ samples}{2}$$

4.2.9.6.3.2 Determine Geometric Grid Spaced Tie-points (See notes #6 and #7)

For descriptions of the format and data stored within the geometric grid, see the Line-of-Sight Projection/Grid Generation Algorithm in Section 4.2.2.

Geometric Space Tie-points Processing steps

1. Read the image extent parameters from the geometric grid.
   INS = input (raw) space number of samples
   INL = input (raw) space number of lines
2. Determine the number of tie-points in the sample and line direction:

$$\text{spacing x} = \frac{\text{INS} - \text{correlation window samples}}{\text{M - 1}}$$

$$\text{spacing y} = \frac{\text{INL} - \text{correlation window lines}}{\text{N - 1}}$$

3. Establish the input (raw) space tie-point locations.
   3.1 For j = 0 to N-2

$$y[j] = \frac{correlation\ window\ lines}{2} + j * spacing\ y$$

   3.2

$$y[N-1] = INL - \frac{correlation\ window\ lines}{2}$$

   3.3 For i = 0 to M-2

$$x[i] = \frac{correlation\ window\ samples}{2} + i * spacing\ x$$

   3.4

$$x[M-1] = INS - \frac{correlation\ window\ samples}{2}$$

4. Project inputs space tie-points locations to output space.
   4.1 For j=N-1
      4.1.1 For i=M-1
         Map the input space tie-point location to the output space using grid mapping coeffcients.
            tie-point location y = $b_0 + b_1 * x[i] + b_2 * y[j] + b_3 * x[i] * y[j]$
            tie-point location x = $a_0 + a_1 * x[i] + a_2 * y[j] + a_3 * x[i] * y[j]$
      Where (See note #7):
      a_n = geometric grid forward sample mapping coefficients for the zero elevation plane retrieved from the resampling grid
      b_n = geometric grid forward line mapping coefficients for the zero elevation plane retrieved from the resampling grid

4.2.9.6.4 Stage 4 - Calculate Individual Point-by-Point Band Displacements

Normalized cross correlation is the standard method used to measure spatial differences between the reference and search windows extracted from the two bands being compared. The normalized cross correlation process helps to reduce any correlation artifacts that may arise from radiometric differences between the two image sources. The correlation process will only measure linear distortions over the windowed areas. By choosing appropriate correlation windows that are well distributed throughout the imagery, non-linear differences between the image sources can be found. Normalized gray scale correlation has the following formula:

$$R(x,y) = \frac{\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2}\left[\left(f(j,i)-\bar{f}\right)\left(g(x+j,y+i)-\bar{g}\right)\right]}{\left[\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2}\left(f(j,i)-\bar{f}\right)^2 \sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2}\left(g(x+j,y+i)-\bar{g}\right)^2\right]^{1/2}}$$

Where:

N = M = Correlation window size in lines and samples

R = correlation surface (N,M) (See note# 10)

F = reference window (N,M)

G = search window (N,M)

$$\bar{f} = \frac{1}{(M+1)(N+1)}\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2} f(j,i)$$

$$\bar{g} = \frac{1}{(M+1)(N+1)}\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2} g(x+j,y+i)$$

Normalized cross correlation will produce a discrete correlation surface (i.e., correlation values at integer x,y locations). A subpixel location associated with the displacement is found by fitting a polynomial around a 3x3 area centered on the correlation peak. The polynomial coefficients can be used to solve for the peak or subpixel location. Once the discrete correlation has been calculated and the peak value within these discrete values has been found, the subpixel location can be calculated:

$$P(y,x) = a_0 + a_1 x + a_2 y + a_3 xy + a_4 x^2 + a_5 y^2$$

Where

P(x,y) is polynomial peak fit

x = sample direction

y = line direction

Set up matrices for a least-squares fit of discrete R(x,y) to x/y locations.

$$\underset{9x1}{\begin{bmatrix} R(-1,-1) \\ R(-1,0) \\ \vdots \\ R(1,0) \\ R(1,1) \end{bmatrix}} = \underset{9x6}{\begin{bmatrix} 1 & -1 & -1 & (-1)*(-1) & (-1)^2 & (-1)^2 \\ 1 & 0 & -1 & (-1)*(0) & (0)^2 & (-1)^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 1 & (1)*(0) & (0)^2 & (0)^2 \\ 1 & 1 & 1 & (1)*(1) & (1)^2 & (1)^2 \end{bmatrix}} \underset{6x1}{[a]}$$

or: [Y] = [X] [a]

Note that R(x,y) is relative to the peak; the total offset will need to have the integer line offset and sample offset added to the subpixel location to have the total measured offset. Solving for the peak polynomial using least squares:

$$[a] = \left([X]^T [X] \right)^{-1} [X]^T [Y]$$

Calculating the partial derivative of P(x,y) in both the x and y directions, setting the partial equations to zero, and solving the partials for x and y, gives the subpixel location within the subpixel 3x3 window.

$$\frac{\partial}{\partial x} P(x, y) = a_1 + a_3 y + 2a_4 x = 0$$
$$\frac{\partial}{\partial y} P(x, y) = a_2 + a_3 x + 2a_5 y = 0$$

Set partial equations equal to zero and solve for x and y:

$$\text{Subpixel x offset} = \frac{2a_1 a_5 - a_2 a_3}{a_3^2 - 4a_4 a_5}$$
$$\text{Subpixel y offset} = \frac{2a_2 a_4 - a_1 a_3}{a_3^2 - 4a_4 a_5}$$

The processing steps subsection provides the steps for mensuration, calculating the total offset measured, and how they fit in the overall procedure.

See the Ground Control Point Correlation Algorithm in Section 4.1.6 for additional description of the normalized gray scale correlation processes.

Least Squares Fine Correlation Method

The band-to-band and image-to-image accuracy characterization algorithms also provide a second, least-squares based correlation method that can be used to measure subpixel image displacements somewhat more reliably than the cross-correlation/peak finding method used for general purpose correlation. This is useful for band registration measurements where the displacements should always be much less than a pixel, and where the quadratic peak finding method can introduce small offset-dependent biases in the measurements. This method requires that the reference and search image windows be the same size and that the offsets to be determined be less than 1 pixel. Since the normalized gray scale correlation algorithm does not work on image windows whose dimensions are not even numbers, this least squares correlation method is invoked if either window dimension is an odd number.

The least-squares correlation method uses the reference and search image window pixels to estimate the sample offset (Δsample), line offset (Δline), gain, and bias adjustments that best match the (Δsample, Δline) shifted and bilinearly interpolated search image to the radiometrically adjusted (1+Δgain, Δbias) reference image. The 3x3 pixel image subwindow surrounding each interior (non-edge) image pixel in the reference and search windows provides one observation for purposes of estimating the four adjustment parameters, using the following model:

$S_0 + S_x * \Delta sample + S_y * \Delta line + S_{xy} * \Delta sample * \Delta line = R_0 * (1+\Delta gain) - \Delta bias$

Where:

$S_0 = S_{i,j}$ = the central pixel in the 3x3 search subwindow centered at (i,j)
$S_x = (S_{i+1,j} – S_{i-1,j})/2$ = slope estimate in the sample direction
$S_y = (S_{i,j+1} – S_{i,j-1})/2$ = slope estimate in the line direction
$S_{xy} = (S_{i+1,j+1} + S_{i-1,j-1} – S_{i+1,j-1} - S_{i-1,j+1})/4$ = rate of slope change
$R_0 = R_{i,j}$ = the reference image pixel corresponding to $S_{i,j}$

This can be reorganized into an observation model for the 4 fit parameters:

$S_x*\Delta sample + S_y*\Delta line - R_0*\Delta gain + \Delta bias = R_0 - S_0 - S_{xy}*\Delta sample*\Delta line$

Or:

$$[S_x \quad S_y \quad -R_0 \quad 1] \begin{bmatrix} \Delta sample \\ \Delta line \\ \Delta gain \\ \Delta bias \end{bmatrix} = \left[R_0 - S_0 - S_{xy}\Delta sample\Delta line\right]$$

[1 x 4] [4 x 1] = [1 x 1] Array sizes

Note that this equation is not linear (since Δsample and Δline appear on the right side) and must be solved iteratively.

Each non-edge pixel generates an observation of this form:

$[X_{i,j}]^T [coef] = [Y_{i,j}]$

Where:

$[X_{i,j}]^T = [S_x \;\; S_y \;\; -R_0 \;\; 1]$ (1x4 matrix)
$[coef] = [\Delta sample \;\; \Delta line \;\; \Delta gain \;\; \Delta bias]^T$ (4x1 matrix)
$[Y_{i,j}] = [R_0 – S_0 – S_{xy} \Delta sample \; \Delta line]$ (1x1 matrix)

Taken together, these observations can be used to compute the best fit, in the least-squares sense, values for the four fit parameters:

$[N] = \Sigma [X_{i,j}] [X_{i,j}]^T$ (4x4 matrix)
$[C] = \Sigma [X_{i,j}] [Y_{i,j}]^T$ (4x1 matrix)
$[coef] = [N]^{-1} [C]$ (4x1 matrix)

The computed values of the fit parameters in [coef] are used to update the $[Y_{i,j}]$ values for each iteration.

The solution procedure is as follows:

1. Verify that the input reference and search windows are the same size and that the window dimensions are both at least 3 pixels.
2. Initialize the least-squares solution normal equations:
    a. Set all 4 elements of the 4x1 constants vector **C** to zero.
    b. Set all 16 elements of the 4x4 normal equation matrix **N** to zero.
    c. Set the normal equation diagonal term corresponding to the gain parameter, **N**[2][2], to $1/\sigma_g^2$, where σ_g is the apriori standard deviation of the gain parameter, set to 0.05 (5%) to limit the magnitude of the gain adjustment.
    d. Set the normal equation diagonal term corresponding to the bias parameter, **N**[3][3], to $1/\sigma_b^2$, where σ_b is the apriori standard deviation of the bias parameter, set to 5 DN to limit the magnitude of the bias adjustment.
    e. Initialize the four adjustment parameter values to zero.
3. Iterate the solution three times. For each iteration:
    a. Loop through the non-edge image pixels, $S_{i,j}$, $R_{i,j}$, in the N_{samp} by N_{line} image windows, where $0 < i < N_{samp}-1$ and $0 < j < N_{line}-1$. For each pixel:
        i. Compute S_0, S_x, S_y, and S_{xy} from the 3x3 search subwindow surrounding the current pixel, using the equations above.
        ii. Compute the right side of the observation equation using R_0 and the current estimates of Δsample and Δline:
            $RHS = R_0 - S_0 - S_{xy} * \Delta sample * \Delta line$
        iii. Add this observation to the normal equations:
            N[0][0] += $S_x * S_x$
            N[0][1] += $S_x * S_y$
            N[0][2] -= $S_x * R_0$
            N[0][3] += S_x
            C[0] += $S_x * RHS$
            N[1][1] += $S_y * S_y$
            N[1][2] -= $S_y * R_0$
            N[1][3] += S_y
            C[1] += $S_y * RHS$
            N[2][2] += $R_0 * R_0$
            N[2][3] -= R_0
            C[2] -= $R_0 * RHS$
            N[3][3] += 1
            C[3] += RHS
    b. Complete the symmetric normal equation matrix:
        N[1][0] = **N**[0][1]
        N[2][0] = **N**[0][2]
        N[2][1] = **N**[1][2]
        N[3][0] = **N**[0][3]
        N[3][1] = **N**[1][3]
        N[3][2] = **N**[2][3]
    c. Solve the normal equations:

$$\mathbf{X} = [\ \Delta sample\ \ \Delta line\ \ \Delta gain\ \ \Delta bias\ ]^T = \mathbf{N}^{-1}\ \mathbf{C}$$

4. Return the results of the final iteration:
   Fit_offset[0] = Δsample
   Fit_offset[1] = Δline
   Diag_Displacement = sqrt(Δsample * Δsample + Δline * Δline)

4.2.9.6.5 Stage 5 - Removing Outliers Using the t-distribution

Once all of the line and sample offsets have been measured and the first level of outlier rejection has been performed, and a check against the maximum allowable offset and the minimum allowable correlation peak takes place, the measurements are further reduced of outliers using a Student-t outlier rejection.

Given a t-distribution tolerance value, outliers are removed within the dataset until all values deemed as “non-outliers” or “valid” fall inside the confidence interval of a t-distribution. The tolerance, or associated confidence interval, is specified per run (or processing flow) and usually lies between 0.9-0.99. The default value is 0.95. The number of degrees of freedom of the dataset is equal to the number of valid data points minus one. The steps involved in this outlier procedure are below. The process listed works on lines and samples simultaneously, calculating statistics independently for each.

Student-t Outlier Rejection Processing Steps

1. Calculate the mean and standard deviation of data for lines and samples (see stage #6).

$$\text{mean offset} = \frac{1}{N}\sum_{i=0}^{N} \text{offset}_i$$

$$\text{standard deviation} = \frac{1}{N-1}\sum_{i=0}^{N} (\text{offset}_i - \text{mean offset})^2$$

   Where:
   N = number of valid offsets measured (above the peak threshold and below the maximum offset)

   Two means and standard deviations are calculated, one for the line direction and one for the sample direction.
2. Find the largest offset and compare it to outlier threshold.
   2.1. Calculate the two-tailed t-distribution (T) value for the current degree of freedom (N-1) and confidence level α.
   2.2. Calculate the largest deviation from the mean allowable for the specified degree of freedom and α:
   $\Delta line = \sigma_{line} * T$
   $\Delta sample = \sigma_{sample} * T$
   Where:
   σ_{line} = standard deviation of valid line offsets

σ_{sample} = standard deviation of valid sample offsets

2.3. Find the valid data point that is farthest from the mean.
max $line_i$ = MAX{ ABS(line offset - mean line offset) }
max $sample_j$ = MAX{ ABS(sample offset - mean sample offset) }
Where:
The maximum is found from all valid offsets
i is the tie-point number of max line
j is the tie-point number of max sample

2.4. If the valid data point that is farthest from the mean is greater than the allowable Δ, then the valid point is flagged as outlier.
if max $line_i$ > Δline or max $sample_j$ > Δsample, then
if(max $sample_j$ / σ_{sample} > max $line_i$ / σ_{line})
tie-point j is marked as an outlier
else
tie-point i is marked as an outlier
else no outliers found

3. Repeat 1 and 2 above until no outliers are found.

4.2.9.6.6 Stage 6 - Calculating Measured Statistics

The mean, standard deviation, minimum, maximum, median, and RMS offset are calculated from the tie-points that pass all outlier criteria: below maximum offset, above peak threshold, and student t-distribution test. The calculation for mean, standard deviation, and RMS are below, where x_i is the measured offset.

mean: $m_x = \frac{\sum_{i=0}^{N-1} x_i}{N}$

standard deviation: $\sigma_x = \sqrt{\frac{\sum_{i=0}^{N-1} (x_i - m_x)^2}{N-1}}$

RMS: $RMS_x = \sqrt{\frac{\sum_{i=0}^{N-1} (x_i)^2}{N}}$

4.2.9.6.6.1 Band Accuracy Assessment Processing Steps

Windows extracted from imagery have the user-entered dimensions, correlation window lines, and correlation window samples. Correlation parameters have been read or set as default values: maximum offset, fit method, correlation peak, fill data range, fill threshold. The bands should be indexed so that the PAN band is always used as a reference to all other bands.

1. For SCA = Number of SCAs to process
1.1. For rband = Number of OLI bands to process

if rband is equal to PAN, use the reduced PAN image file

1.2. For sband = rband + 1 to Number of OLI bands to process

1.3. For index = Number of tie-points to process

1.3.1. Read the current tie-point chip and tie-point location x,y

Set the tie-point flag to unsuccessful

1.3.2. Extract the sband window (of imagery) at tie-point location x,y

1.3.3. Extract the rband window (of imagery) at tie-point location x,y

1.3.4. Count the number of pixels in rband window that is within fill range.

count = 0

For i=0 to number of pixels in correlation window

If rband pixel is > fill min and rband pixel is < fill max

count++

1.3.5. Check the number of rband pixels counted against the fill threshold/percentage.

$$if\ \frac{count}{correlation\ window\ size} > fill\ threshold$$

increment index to next tie-point location

else

continue

1.3.6. Count the number of pixels in the sband window that is within the fill range.

count = 0

For i=0 to number of pixels in the correlation window

If sband pixel is > fill min and sand pixel is < fill max

count++

1.3.7. Check the number of sbands pixels counted against the fill threshold/percentage.

$$if\ \frac{count}{correlation\ window\ size} > fill\ threshold$$

increment the index to next tie-point location

else

continue

1.3.8. Perform normalized gray scaled correlation between the rband and sband windowed images, calculating correlation surface *R* (See Stage 4 and notes #9 and #10).

1.3.9. Find the peak within the correlation surface

Max = *R*(0,0)

For i=0 to correlation window number of lines -1

For j=0 to correlation window number of samples -1

If *R*(i,j) > max then

Max = *R*(i,j)

line offset = i

sample offset = j

1.3.10. Check the correlation peak against the threshold
if max > peak threshold
continue
else
set the tie-point flag to outlier and choose next tie-point
1.3.11. Measure the subpixel peak location (see stage #4)
Δsub-line
Δsub-sample
1.3.12. Calculate the total pixel offset
total line offset = line offset + Δsub-line
total sample offset = sample offset + Δsub-sample
1.3.13. Check the offset against the maximum displacement offset

$$\text{total displacement} = \sqrt{(\text{total line offset})^2 + (\text{total sample offset})^2}$$

if (total displacement > maximum displacement)
Set the tie-point flag to outlier and choose the next tie-point
Else
Set the tie-point flag to valid

1.4. Store SCA and band combination (rband-to-sband) tie-point mensuration information, correlation success, and offsets measured. See Table 4-24.
2. For SCA = 1 to Number of SCAs to process
2.1. For band combination = 1 to Number of band combinations
2.1.1. Perform t-distribution outlier rejection (See stage #5).
2.2. Store the SCA and band combination final individual tie-point information and outlier flag. See Table 4-25.
3. For SCA = 1 to Number of SCAs to process
3.1. For band combination = 1 to Number of band combinations
3.1.1. Calculate the mean, minimum, maximum, median, standard deviation, and root mean squared offset.
3.1.2. Store the SCA and band combination statistics. See Table 4-26.
4. Perform trending if trending flag is set to yes
4.1. Check the results against the trending thresholds
For each band of each SCA
if measured Standard Deviation > trending threshold
exit trending
If there are no Standard Deviation > trending thresholds perform trending

4.2.9.6.7 Output files

The output files listed below for the BRAA are made generic so that the same format can be used elsewhere. Some fields, such as latitude, longitude, and elevation, may not apply to the application and would be filled with zeros or nominal values.

All output files contain a standard header. This standard header is at the beginning of the file and contains the following:

- Date and time the file was created
- Spacecraft and instrument pertaining to measurements

- Off-nadir (roll) angle of the spacecraft/instrument
- Acquisition type
- Report type (band-to-band)
- Work order ID of process (left blank if not applicable)
- WRS path/row
- Software version that produced the report
- L0R image file name

The data shown in Table 4-24, Table 4-25, and Table 4-26 is stored in the database. The statistics stored per band per SCA will be used for trending analysis of the band registration accuracy of the OLI instrument. Results produced through a time-series analysis of this data stored, over a set time interval or multiple image files, will determine if new LOS Legendre coefficients will need to be generated from the OLI Band-to-Band Calibration Algorithm (See OLI Band-to-Band Calibration Algorithm in Section 4.2.10 for details). These statistics may also be needed for providing feedback to the L8/9 user community about the band registration of the L8/9 products generated.

Field	Description
Date and time	Date (day of week, month, day of month, year) and time of file creation.
Spacecraft and instrument source	Landsat 8/9 and OLI/OLI-2 (TIRS/TIRS-2 if applicable)
Processing Center	EROS Data Center SVT
Work order ID	Work order ID associated with processing (blank if not applicable)
WRS path/row	WRS path and row (See note #11)
Software version	Software version used to create report
Off-nadir angle	Off-nadir roll angle of processed image file (See note #12)
Acquisition Type	Earth viewing or Lunar
L0R image file	L0R image file name used to create L1TP
Processed image file name	Name of L1TP used to create report
Reference bands	Reference bands used in band assessment
Search bands	Search bands used in band assessment
Heading for individual tie-points	One line of ASCII text defining individual tie-point fields.
For each tie-point:	
Tie point number	Tie-point index/number in total tie-point list
Reference line	Tie-point line location in reference image (band)
Reference sample	Tie-point sample location in reference image (band)
Reference latitude	Tie-point latitude location
Reference longitude	Tie-point longitude location
Reference elevation	Elevation of tie-point location (see note #13)
Search line	Tie-point line location in search image
Search sample	Tie-point sample location in search image
Delta line	Measured offset in line direction
Delta sample	Measured offset in sample direction
Outlier flag	1=Valid, 0=Outlier
Reference band	Reference band number
Search band	Search band number
Reference SCA	SCA number that reference window was extracted

Field	Description
Search SCA	SCA number that search window was extracted
Search image	Name of search image
Reference image	Name of reference image

Table 4-24. Band Registration Accuracy Assessment Data File

Field	Description
Date and time	Date (day of week, month, day of month, year) and time of file creation.
Spacecraft and instrument source	Landsat 8/9 and OLI/OLI-2 (TIRS/TIRS-2 if applicable)
Processing Center	EROS Data Center SVT
Work order ID	Work order ID associated with processing (blank if not applicable)
WRS path/row	WRS path and row (See note #11)
Software version	Software version used to create report
Off-nadir angle	Off-nadir pointing angle of processed image file (See note #12)
Acquisition Type	Earth viewing or Lunar
L0R image file	L0R image file name used to create L1TP
Processed image file name	Name of L1TP used to create report
Number of records	Total number of tie-points stored in file
Heading for individual tie-points	One line of ASCII text defining individual tie-point fields.
For each band combination	
Combination header	Number of points in combination, reference band number, search band number.
For each tie-point:	
Tie point number	Tie-point index/number in total tie-point list
Reference line	Tie-point line location in reference image (band)
Reference sample	Tie-point sample location in reference image (band)
Reference latitude	Tie-point latitude location
Reference longitude	Tie-point longitude location
Reference elevation	Elevation of tie-point location
Search line	Tie-point line location in search image
Search sample	Tie-point sample location in search image
Delta line	Measured offset in line direction
Delta sample	Measured offset in sample direction
Outlier flag	1=Valid, 0=Outlier
Correlation coef	Correlation coefficient for tie point correlation
Reference band	Reference band number
Search band	Search band number
Reference SCA	SCA number that reference window was extracted from
Search SCA	SCA number that search window was extracted from
Search image	Name of search image
Reference image	Name of reference image

Table 4-25. Band Registration Accuracy Assessment Residuals File

Field	Description
Date and time	Date (day of week, month, day of month, year) and time of file creation.
Spacecraft and instrument source	Landsat 8/9 and OLI/OLI-2 (TIRS/TIRS-2 if applicable)
Processing Center	EROS Data Center SVT
Work order ID	Work order ID associated with processing (blank if not applicable)
WRS path/row	WRS path and row (See note #12)
Software version	Software version used to create report
Off-nadir angle	Off-nadir pointing angle of processed image file (See note #13)
Acquisition Type	Earth viewing or Lunar
L0R image file	L0R image file name used to create L1TP
Processed image file name	Name of L1TP used to create report
t-distribution threshold	Threshold used in t-distribution outlier rejection
For each band combination of each SCA processed	
Reference band	Reference band of statistics
Search band	Search band of statistics
SCA	SCA number of statistics
Total tie-points	Total number of tie-points for band combination of SCA
Correlated tie-points	Number of tie-points that successfully correlated for band combination of SCA
Valid tie-points	Total number of valid tie-points for band combination of SCA after all outlier rejection has been performed
For both line and sample direction:	All statistics are given in terms of pixels
Minimum offset	Minimum offset within all valid offsets
Mean offset	Mean offset of all valid offsets
Maximum offset	Maximum offset within all valid offsets
Median offset	Median offset within all valid offsets
Standard deviation	Standard deviation of all valid offsets
Root-mean-squared	Root mean squared offset of all valid offsets

Table 4-26. Band Registration Accuracy Assessment Statistics Output File

4.2.9.6.8 Assessing Band Registration (Accessing Statistics Stored in Database)

The geometric CalVal team will need to access the Band Accuracy Assessment statistics stored in the database. Delineation, or essentially data base querying, will take place by the following or a combination of the following:

- Date range of the image acquisition or processing date
- By SCA number
- By band number
- By acquisition type (Nadir, off-nadir, Lunar)
- By geographic location of image extent

At a minimum, access to the Band Accuracy Assessment statistics is needed. Simple tools, such as an SQL queries, would be beneficial to the geometric CalVal team, but are not absolutely necessary, as they could be developed later through other means.

4.2.9.7 Notes

Some additional background assumptions and notes include the following:

1. Correlation parameters, minimum correlation peak, and maximum offset are stored and retrieved from the CPF.
2. Options need to be available for generating statistics; scene statistics, individual bands per SCA, SCA summary, band summary. These statistics would be provided to the user as summary statistics to be provided as image quality assessment to the user community.
3. There will need to be a set of criteria, based on calculated statistics, as to whether trending should be performed. These criteria would be provided to avoid having garbage stored in the database. Any values needed in determining whether the criteria have been met for trending would be stored and retrieved from the CPF. There would be one threshold per band per SCA. The criteria to check for trending are shown in Section 4.1 of the Band Accuracy Assessment Processing steps section.
4. Band Accuracy statistics stored within the database will be accessed for analysis.
    - *Accessed according to a specific date range.*
    - *Accessed according to a specific band or SCA.*
    - *Accessed according to a specific geographic location.*
    - *Accessed according to acquisition type (nadir, off-nadir, lunar).*
    - This data accessed will be retrieved and stored within a comma-delimited file. The methodology used to access the database could be an SQL script. This SQL query code could be developed either by the IPE or outside of the IPE.
5. Data stored within the database will be accessed for time series analysis.
    - Data would be pulled out by scene/SCA band pairs for a user-specified time.
    - Statistics over multiple scenes would be calculated per SCA and/or per band, and then combined into the SCA and/or band average statistics.
    - Results could be compared to the band registration spec. These results could serve as triggers to other events, i.e., new CPF generation and testing.
    - Results could be used to verify conformance with product specifications.
    - These calculations could be performed within the methodology used to access the data from the database (SQL script).
6. Tie-point locations could also be stored and used as projection Y and X coordinates. The appropriate conversions must be done when converting between projection coordinates and line and sample locations when extracting image windows between bands. This transformation should also include any rotation due to path-orientated projections.

7. The code uses a library call that maps any input point with a given elevation to output space. For BRAA, the elevation for the mapping point is set to zero. Since the reference and search output space are the same for BRAA, the output line/sample in output reference space should be the line/sample in output search space.
8. The c and d parallax coefficients are needed for each band or each SCA for every grid cell point. Therefore, if the coefficients were stored as arrays stacked by grid column, and then grid row for a particular input pixel that fell within grid cell column N and grid cell row M, the c and d coefficients needed for that pixel would be indexed by: index = (M * number of grid columns + N) * 2. The factor of 2 is because the parallax odd/even effects are mapped as linear; therefore, 2 coefficients are stored for each the odd and even pixels of a grid cell.
9. The gray scale correlation process, or surface, can be implemented using a Fast Fourier Transform (FFT).
10. The correlation surface could be smaller than the search window, depending on the search area or maximum offset.
11. Any kind of "non-WRS" collect other than solar and lamp, such as lunar, should have 000/000 listed as the path/row.
12. The pointing angle for lunar acquisitions would be 0.0.
13. This tie-point residual file structure is also used for the image registration accuracy characterization algorithm, so it includes fields that are not required for both algorithms. An example is the elevation field, which is set to 0 for this algorithm.
14. The correlation result fit method defines the algorithm used to estimate the correlation peak location to subpixel accuracy. Only the quadratic surface fitting method described in this algorithm description is supported in the baseline algorithm. The Least-Squares Correlation technique does not use the surface-fitting method; for the gray scale correlation technique, the peak fitting method still applies.

4.2.10 OLI Band-to-Band Calibration Algorithm

4.2.10.1 Background/Introduction

The Band-to-Band Calibration (B2BCal), or Band Calibration, algorithm estimates improved values for band placement within each Sensor Chip Assembly (SCA) of the OLI instrument. Adjustments are made relative to the PAN band, or in other words, the PAN band serves as the reference for all other bands.

The B2B calibration takes the Band Accuracy Assessment residuals file, which represents displacements with respect to the product output projection space, maps the residuals back into displacements with respect to the focal plane, and then performs a Least-Squares (LSQ) fit between the focal plane residuals to determine updates to the OLI band Legendre LOS polynomial coefficients. The least squares fit results represent updates needed to adjust the existing Legendre LOS coefficients. These updates can be used to produce new Legendre LOS coefficients for the CPF.

Other than a few high altitude calibration sites where the ground is visible in the cirrus band, most Earth-viewing acquisitions will be used to calibrate all bands except the cirrus. A lunar acquisition or a high-elevation Earth target acquisition will be used to calibrate the cirrus band.

4.2.10.2 Dependencies

The OLI B2B calibration algorithm assumes that a cloud-free nadir-viewing L1TP image has been generated and the resampled DEM used to create the L1TP is available. The LOS Model Creation (4.2.1) and LOS Projection/Gridding (4.2.2) algorithms for the L1TP will be assumed to have been executed, and the corresponding output files available. The L1TP image needs to be in an SCA-separated format and either in a SOM or UTM path-oriented projection. For best results, the DOQ control and best available DEM should be used in generating the L1TP. The accuracy of the precision solution should have post-fit residuals below the recommended threshold, the solution should have used an adequate number of control points, and the distribution of the control should be well distributed throughout the imagery. The Band Registration Accuracy Assessment, or B2BChar algorithm, of Section 4.2.9 is assumed to have been run on the L1TP image, successfully producing a Band Accuracy Assessment residuals file. This Band Accuracy Assessment residuals file, along with the CPF, geometric LOS model, the co-registered DEM, and the geometric LOS resampling grid, are used as inputs to the Band Calibration algorithm.

4.2.10.3 Inputs

The B2B calibration algorithm uses the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs	Algorithm Inputs
OLI resampling grid	ODL
DEM	ODL
OLI CPF file name	ODL
Along-track IFOV	CPF/LOS-model
Minimum points	ODL
Number of Legendre Coefficients	ODL (See note #6)
OLI Line-of-Sight model	ODL
B2B residuals file	ODL
Band calibration report file	ODL
Trend flag	ODL
Flag for CPF group creation (see note #3)	ODL
Flag for individual tie-point listing	ODL
CPF effective dates (begin and end)	ODL
L0R ID (for trending)	ODL
Work Order ID (for trending)	ODL

4.2.10.4 Outputs

B2B calibration report file (See note #1 and Table 4-27)
Legendre LOS CPF group
B2B calibration trending
L0R/L1R ID

Work Order ID
WRS Path/Row
B2B calibration post and pre fit residuals
New SCA line-of-sight parameters

4.2.10.5 Options

Trending on/off switch
fits LOS group within OLI CPF.

4.2.10.6 Procedure

Band calibration uses the residuals measured during the Band Registration Accuracy Algorithm (see Section 4.2.9) to determine updates to the Legendre LOS coefficients (See Line-of-Sight Model Creation Algorithm 4.2.1). The band calibration process involves taking the residuals from Band Registration Accuracy Assessment, measured in output space, mapping them into input space angular deltas in terms of along- and across-track LOS angles, and performing a least-squares fit of the input space LOS angle deltas to a set of 2^{nd} order Legendre polynomial correction coefficients. The correction polynomials calculated represent updates to the original LOS Legendre polynomial coefficients. New Legendre LOS coefficients can be found by combining the correction coefficients with the original coefficients.

Due to the differences in viewing geometry between bands within a SCA, along with the differences in viewing geometry between SCAs, the effects due to relief displacement must be taken into account during band calibration. To account for relief displacement during B2B calibration, a DEM is required. The resampling grid and LOS model is also required during B2B calibration. The resampling grid, the corresponding detector's IFOV, and the LOS model's Legendre coefficients are used to map the residuals from output space to angular differences in input space.

A least-squares fit is done on all requested bands and SCAs using the band-to-band tie point measurements from all band-pair combinations for a single SCA at a time. Requested bands and SCAs to process are based on the bands and SCAs present within the Band Registration Accuracy Assessment residuals file.

4.2.10.6.1 Stage 1 - Data input

The data input stage involves loading the information required to perform the band calibration. Input file names are needed for: geometric LOS resampling grid, LOS model, Band Registration Accuracy Assessment results (B2B residuals file), output band calibration report file name, and the L1TP DEM file name. Further input parameters are the effective begin and end dates of the new LOS Legendre coefficients calculated, trending flag, CPF group creation flag, and individual tie-point reporting. Once the file names for the input data needed are retrieved, the files can be opened and read.

Get ODL Parameters
Reads the parameters from the input ODL parameter file. This process was modified from the ALIAS heritage version to handle new inputs: minimum points, flag for CPF group creation, CPF effective dates, and flag for reporting individual tie-point results. The minimum points variable ensures that the normal matrix contains a minimum number along its diagonal to zero out any omitted bands. Rather than being removed from the solution, the offsets for omitted bands are set to zero with a weight equal to the minimum number of points.

Read Band-to-Band Residual File
Reads the band accuracy assessment residuals file.

Read DEM
Reads the DEM file into the IMAGE data structure.

Read OLI LOS Model
Reads the OLI geometric/LOS model.

Read LOS Geometric Grid
Reads the OLI LOS resampling grid.

4.2.10.6.2 Stage 2 - Setup Least Squares Matrices and Solve

For each input SCA, every residual for each input band combination that is not an outlier is mapped back to input space. These input space mappings for each point calculate the adjustments needed to align the lines of sight, projected from the focal plane, for the two bands of the combination. This mapping procedure is described in more detail below. Once all of these residuals are mapped back to the focal plane and stored within the LSQ matrices, adjustments to the Legendre LOS model can be calculated. PAN band residuals must be scaled by a factor of 2 to account for the resolution differences between the PAN band and the multispectral bands, and the fact that the PAN residuals were measured in an image that had been resolution reduced to match the multispectral bands.

The matrices defining the calibration process takes the following form:

$$[A][coeff] = [Y]$$

The matrices [*A*] and [*Y*] shown above correspond to one tie-point measurement. The matrix [*coeff*] are the unknown adjustments to the Legendre LOS coefficients, the matrix [*A*] contain the Legendre coefficient multipliers for the band combination corresponding to that one measurement, and the [*Y*] matrix contains the input space residuals for that one measurement. For one measurement, the matrices have the following dimensions:
[coeff] = (2 * Number of Legendre * Number of bands) x 1 = M x 1
[A] = 2 x (2 * Number of Legendre * Number of bands) = 2 x M
[Y] = 2 x 1

$$[coeff] = \begin{bmatrix} a_{b1,0} \\ a_{b1,1} \\ a_{b1,2} \\ b_{b1,0} \\ b_{b1,1} \\ b_{b1,2} \\ a_{b2,0} \\ a_{b2,1} \\ a_{b2,2} \\ b_{b2,0} \\ \vdots \\ b_{b9,0} \\ b_{b9,1} \\ b_{b9,2} \end{bmatrix}$$

Where:

$a_{bi,j}$ = Legendre coefficient j for line direction (along-track) for band i
$b_{bi,j}$ = Legendre coefficient j for sample direction (across-track) for band i
j = 0, 1, 2 or the Number of Legendre coefficients to solve.
i = 1,2,...,9 (Number of OLI bands)

A 2x1 matrix pertaining to one residual measurement can be defined as follows:

$$[Y] = \begin{bmatrix} \Delta line \\ \Delta sample \end{bmatrix}$$

Where:

$\Delta line$ = input space residual in line direction (angular)
$\Delta sample$ = input space residual in sample direction (angular)

The input space residuals are calculated by finding the nominal (search) LOS in input space and the measured (search + measured offset) LOS in input space. These lines of sight are found by mapping the output space line and sample locations to input space line and sample locations using the LOS geometric resampling grid (See OLI Resampling Algorithm 4.2.4) and then using the LOS model (see LOS Model Line-of-Sight Creation Algorithm 4.2.1) to convert the input space locations to lines of sight. These input space nominal and measured locations are also used to construct the Legendre coefficient multipliers.

The design matrix [*A*] for one residual measurement is then:

$$[A_{nik}][coeff] = [Y_n]$$

$$[A]=$$
$$\begin{bmatrix} 0 & \dots & 0 & -rl_{n,i,0} & \dots & -rl_{n,i,2} & 0 & 0 & 0 & 0 & \dots & 0 & sl_{n,k,0} & \dots & sl_{n,k,2} & 0 & 0 & 0 & 0 & \dots \\ 0 & \dots & 0 & 0 & 0 & 0 & -rl_{n,i,0} & \dots & -rl_{n,i,2} & 0 & \dots & 0 & 0 & 0 & 0 & sl_{n,k,0} & \dots & sl_{n,k,2} & 0 & \dots \end{bmatrix}$$

Where:
$rl_{n,i,j}$ = reference band i Legendre polynomial
$sl_{n,k,j}$ = search band k Legendre polynomial
j = 0, 1, 2 or the Number of Legendre coefficients to solve
n = tie-point number

These matrices define one observation. A sequence of observations can be summed to define the normal equations for a set of coefficients that can be used to update the OLI LOS Legendre coefficients:

$$[N] = \sum A_{nik}^{T} W_{ik}^{-1} A_{nik}$$
$$[L] = \sum A_{nik}^{T} W_{ik}^{-1} Y_{nik}$$

Where [*N*] and [*L*] are summed over all *n* for all *i, k* band combinations. *W* is a weight matrix that is currently set to the same weight for all observations.

Since all of the tie-point observations involve band differences, the solution lacks an absolute reference. To stabilize the solution, a constraint observation is added to provide such a reference. This additional observation is required for the PAN band and represents an offset of zero for each direction (line and sample) of the PAN band. This allows the PAN band to be used as a reference, and all other bands are then registered to it.

$$[A_{00}] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & \dots & 0 & 0 & 0 \end{bmatrix}$$

$$[Y_{00}] = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Where the PAN band line and sample Legendre polynomial coefficients are the first six columns of the [A] observation matrix.

The bands that are not to be used in the solution process are removed by setting the corresponding diagonal elements of normal matrix [N] to the minimum number of points (given as an input value). The solution for a new set of Legendre coefficients is then:

$$[coeff] = [N]^{-1}[L]$$

Band Calibration Processing Steps

Note: Array indexes are zero-relative.

nLeg = Number of Legendre update coefficients to solve (1, 2, 3 valid options).
Matrix indexes are zero relative

Solve for New Focal Plane Parameters

Loop on each valid tie-point for each SCA and each band combination. Calculate LOS errors to determine per tie-point adjustment needed to LOS. Assimilate normal matrices and solve for updates needed to Legendre LOS, calculate new Legendre LOS based on updates from least-squares-solution. Calculate post and pre-fit statistics.

Calculate Line of Sight Angular Errors

Calculate the delta input line and sample LOS needed for LSQ. Read the elevation for tie-point, map the tie-point to input space, and find adjustments needed between search and reference LOS vectors.

1. Initialize parameters

$$[W] = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix}$$

Where $\sigma^2 = 16$

2. For each SCA to process

Initialize pre-fit statistics variables
pre-fit sum line = 0
pre-fit sum sample = 0
pre-fit sum line^2 = 0
pre-fit sum sample^2 = 0
Initialize LSQ matrices to zero
[N] = [0]
[L] = [0]

2.1 For each band combination present
rband = reference band
sband = search band
2.1.1 For each tie-point
2.1.2 Calculate the reference line, sample location and search adjusted line, sample location.
rline = tie-point reference line location
rsamp = tie-point reference sample location
sline = tie-point search line location + line offset measured
ssamp = tie-point search sample location + sample offset measured

Note: sline, ssamp is the adjusted (or true) search location.

Note that rline, rsamp, sline, ssamp are output space pixel locations.

2.1.3 Set rband and sband to zero-relative

rband = rband - 1

sband = sband – 1

Map residuals to input space (focal plane space)

2.1.4 Find the elevation for the reference and sample locations

relev = elevation at rline,rsamp

selev = elevation at sline,ssamp

2.1.5 Map rline,rsamp and sline,ssamp to input space using 3d_ols2ils (See Note #2) and the search OLI band resampling grid.

(riline,risamp) = 3d_ols2ils(search_grid, relev, rline, rsamp)

(siline,sisamp) = 3d_ols2ils(search_grid, selev, sline, ssamp)

Where

riline, risamp is the input space location of reference tie-point location.

siline, sisamp is the input space location of adjusted search tie-point location.

search_grid is the OLI resampling grid for the search band.

Note: Search band grid is used for mapping both the adjusted search (siline,sisamp) and the reference locations.

2.1.6 Calculate the Legendre normalized detector location

$$\text{rnorm} = \frac{2.0 * \text{risamp}}{\text{number detectors in SCA} - 1} - 1$$

$$\text{snorm} = \frac{2.0 * \text{sisamp}}{\text{number detctors in SCA} - 1} - 1$$

rnorm = normalized reference detector

snorm = normalized adjusted search detector

2.1.7 Calculate the reference and search along- and across-track LOS.

$$nom\_sear\_x = coef\_x_{s,0} + coef\_x_{s,1} * rnorm + coef\_x_{s,2} * (1.5 * rnorm^2 - 0.5)$$

$$nom\_sear\_y = coef\_y_{s,0} + coef\_y_{s,1} * rnorm + coef\_y_{s,2} * (1.5 * rnorm^2 - 0.5)$$

$$sear\_x = coef\_x_{s,0} + coef\_x_{s,1} * snorm + coef\_x_{s,2} * (1.5 * snorm^2 - 0.5)$$

$$sear\_y = coef\_y_{s,0} + coef\_y_{s,1} * snorm + coef\_y_{s,2} * (1.5 * snorm^2 - 0.5)$$

Where

nom_sear_x, nom_sear_y = reference along- and across-track angles

sear_x, sear_y = search along- and across-track view angles

coef_x$_{s,n}$ = search Legendre along-track coefficients

coef_y$_{s,n}$ = search Legendre across-track coefficients

2.1.8 Determine LOS vectors

sear_z = 1.0

$$m = \sqrt{sear\_x^2 + sear\_y^2 + sear\_z^2}$$

$$sear\_x = \frac{sear\_x}{m}$$

$$sear\_y = \frac{sear\_y}{m}$$

$$sear\_z = \frac{sear\_z}{m}$$

nom_sear_z = 1.0

$$m = \sqrt{nom\_sear\_x^2 + nom\_sear\_y^2 + nom\_sear\_z^2}$$

$$nom\_sear\_x = \frac{nom\_sear\_x}{m}$$

$$nom\_sear\_y = \frac{nom\_sear\_y}{m}$$

$$nom\_sear\_z = \frac{nom\_sear\_z}{m}$$

2.1.9 Calculate the LOS errors

2.1.9.1 Determine the effective line-of-sight IFOV

2.1.9.1.1 Map the input search pixel location, line and sample, to output space.

sline = search line location

ssamp = search sample location

elevation = elevation for location sline, ssamp

Calculate the elevation planes bounding the current elevation.

$$zplane = \frac{elevation}{grid\ \ z\ spacing} + grid\ \ zero\ \ plane$$

*elev0 = grid z spacing * (zplane – grid zero plane)*

elev1 = elev0 + grid z spacing

Calculate the cell index, row and column, for the search line and sample location and zplane.

row = sline / grid cell line spacing

column = ssamp / grid cell sample spacing

*cell index0 = nrows * ncols * zplane + row * ncols + column*

Where:

grid z spacing = elevation difference between two grid planes

ncols = number of grid cell columns

nrows = number of grid cell rows

Calculate the output space line, sample location for the input space search line, sample location, and zplane.

$a_{0,1,2,3}$ = grid sample location forward mapping coefficients for cell index0

$b_{0,1,2,3}$ = grid line location forward mapping coefficients for cell index0

*lms = sline * ssamp*
*osamp0 = a_0 + a_1 * ssamp + a_2 * sline + a_3 *lms*
*oline0 = b_0 + b_1 * ssamp + b_2 * sline + b_3 * lms*

Calculate the cell index, row and column, for the search line and sample location, and zplane +1.

*cell index1 = nrows * ncols * (zplane + 1.0) + row * ncols + column*

Calculate the output space line, sample location for the input space search line, sample location, and zplane+1.

$a_{0,1,2,3}$ = grid sample location forward mapping coefficients for cell index1
$b_{0,1,2,3}$ = grid line location forward mapping coefficients for cell index1

*lms = sline * ssamp*
*osamp1 = a_0 + a_1 * ssamp + a_2 * sline + a_3 *lms*
*oline1 = b_0 + b_1 * ssamp + b_2 * sline + b_3 * lms*

Calculate the output space line, sample location for the input space search line, sample location, and elevation.

w0 = (elev1 – elevation) / (elev1 – elev2)
w1 = (elevation – elev0) / (elev1 – elev2)

*$osamp_n$ = osamp0 * w0 + osamp1 * w1*
*$oline_n$ = oline0 * w0 + oline1 * w1*

2.1.9.1.2 Map the input location ssamp, sline+1.0 to output space $osamp_{n+1}$,$oline_{n+1}$ (repeat step 2.1.9.1.1 for input location ssamp,sline+1)
2.1.9.1.3 Determine the change in output space between input locations (ssamp, sline) and (ssamp, sline+1.0)
dline = $oline_n$ – $oline_{n+1}$
dsamp = $osamp_n$ = $osamp_{n+1}$

$$\text{distance} = \sqrt{\text{dline*dline} + \text{dsamp*dsamp}}$$

2.1.9.1.4
If Earth acquisition, calculate the LOS distance to the target

Calculate the output latitude and longitude for the search line and sample (see the Forward Model section of LOS Projection Ellipsoid & Terrain).

Calculate the time for the current search line and sample (see Section a.1 in the Forward Model section of LOS Projection Ellipsoid & Terrain).

Calculate the satellite position for the current search line and sample time (see Section a.4 in the Forward Model section of LOS Projection Ellipsoid & Terrain).

Calculate the target vector (see Section a.7 in the Forward Model section of LOS Projection Ellipsoid & Terrain).

LOS x coordinate = target x coordinate - satellite x coordinate
LOS y coordinate = target y coordinate - satellite y coordinate
LOS z coordinate = target z coordinate - satellite z coordinate

length = sqrt(LOS x * LOS x + LOS y * LOS y + LOS z * LOS z)

$IFOV_{along}$ = (output pixel size * distance) / length

If lunar acquisition

$IFOV_{along}$ = output pixel size * distance
2.1.9.1.5 Calculate the delta input in radians

$$\Delta line = \frac{sear\_x}{sear\_z} - \frac{nom\_sear\_x}{nom\_sear\_z} + (siline - riline) * IFOV_{along}$$

$$\Delta samp = \frac{sear\_y}{sear\_z} - \frac{nom\_sear\_y}{nom\_sear\_z}$$

2.1.10 Create the matrices to sum with [N] and [L].
2.1.10.1 Calculate the Legendre generating polynomial coefficients for search and reference:
sl_0 = 1.0
if(*nLeg* >= 2) sl_1 = snorm
if(*nLeg* == 3) sl_2 = 1.5 * snorm^2 - 0.5
rl_0 = 1.0
if (*nLeg* >= 2) rl_1 = rnorm
if(*nLeg* == 3) rl_2 = 1.5 * rnorm^2 - 0.5
Note: If the number of Legendre coefficients in the solution is less than 3, the corresponding sl_n and rl_n will be omitted.
2.1.10.2 Initialize [A] to zero and then set [A] indexes to sl_n and rl_n.
A[0][Number Legendre * sband + n] = sl_n
A[1][Number Legendre * sband + n] = sl_n
A[0][Number Legendre * rband + n] = -rl_n
A[1][Number Legendre * rband + n] = -rl_n
Where: n = 0 … *nLeg* -1
[A] = 0 elsewhere
2.1.10.3 Set [Y] according to the input space deltas measured and sum pre-fit statistics.
2.1.10.3.1 Store deltas in [Y]

Y[0][0] = Δ*line*
Y[1][0] = Δ*samp*

2.1.10.3.2 Sum statistics.

pre-fit sum line = pre-fit sum line + Δ*line*
pre-fit sum sample = pre-fit sum sample + Δ*sample*
pre-fit sum line^2 = pre-fit sum line^2 + Δ*line*^{2}
pre-fit sum sample^2 = pre-fit sum sample^2 + Δ*sample*^{2}

2.1.10.4 Create matrices to add to normal matrices.

$[A_{tie\text{-}point}] = [A]^T[W][A]$
$[Y_{tie\text{-}point}] = [A]^T[W][Y]$

2.1.10.5 Sum N and L matrices

$[N] = [N] + [A_{tie\text{-}point}]$
$[L] = [L] + [Y_{tie\text{-}point}]$

2.2 Set the minimum points for bands to omit from processing.

Eliminate the observations for the omitted band:
oband = band to omit - 1 (from earlier, bands are 1-relative)
$[N]_{g+n,i} = 0$

Where g = *nLeg* * 2 * oband
n = 0 … 2**nLeg* - 1
i = 0 … *nLeg* * 2 * Number of Bands - 1
$[N]_{i,g+n} = 0$

Where g = *nLeg* * 2 * oband
n = 0 … 2**nLeg* - 1
i = 0 … *nLeg* * 2 * Number of Bands - 1
$[N]_{g+n,g+n}$ = Minimum Points

Where g = *nLeg* * 2 * oband
n = 0 … 2**nLeg* - 1
$[L]_{g+n} = 0$

Where g = *nLeg* * 2 * oband
n = 0 … 2**nLeg* - 1

2.3 Solve for delta Legendre coefficients.

$$[\Delta coeff] = [N]^{-1}[L]$$

2.4 Calculate new Legendre coefficients.

$$\text{new along coeffs}_{\text{SCA,band,n}} = \text{previous along coeffs}_{\text{SCA,band,n}} + \sum_{i=0}^{(nLeg*NBAND)} \sum_{j=0}^{nLeg} \left[\Delta coeff_{i+j}\right]$$

$$\text{new across coeffs}_{\text{SCA,band,n}} = \text{previous across coeffs}_{\text{SCA,band,n}} = \sum_{i=nLeg}^{(nLeg*NBAND)} \sum_{j=0}^{nLeg} \left[\Delta coeff_{i+j}\right]$$

i = 0, *nLeg* , 2 * *nLeg*,…, NBANDS * *nLeg*
n, j = 0 … *nLeg* - 1

4.2.10.6.3 Stage 3 - Calculate Pre- and Post-fit Residuals

Compute Post-Fit Statistics

Calculate pre- and post-fit statistics.

1. For each SCA calculate residuals

Initialize post-fit statistics variables

post-fit sum line = 0

post-fit sum sample = 0

post-fit sumsq line = 0

post-fit sumsq sample = 0

2. For each band combination

2.1 Perform steps 2.1.1 - 2.1.10 from stage 2.

2.2 Calculate the adjusted reference and search line/sample locations

riline' = $\Delta coeff_{sca,rband,0}$

if(*nLeg* >=2) riline' = riline' + rnorm * $\Delta coeff_{sca,rband,1}$

if(*nLeg* = = 3) riline' = riline' + (1.5 * $rnorm^2$ - 0.5) * $\Delta coeff_{sca,rband,2}$

risamp' = $\Delta coeff_{sca,rband,0}$

if(*nLeg* >= 2) risamp' = risamp' + rnorm * $\Delta coeff_{sca,rband,1}$

if(*nLeg* = = 3) risamp' = risamp' + (1.5 * $rnorm^2$ - 0.5) * $\Delta coeff_{sca,rband,2}$

siline' = $\Delta coeff_{sca,rband,0}$

if(*nLeg* >= 2) siline' = siline' + snorm * $\Delta coeff_{sca,sband,1}$

if(*nLeg* = = 3) siline' = siline' + (1.5 * $snorm^2$ - 0.5) * $\Delta coeff_{sca,sband,2}$

sisamp' = $\Delta coeff_{sca,sband,0}$

if(*nLeg* >= 2) sisamp' = sisamp ' + snorm * $\Delta coeff_{sca,sband,1}$

if(*nLeg* = = 3) sisamp' = sisamp' + (1.5 * $snorm^2$ - 0.5) * $\Delta coeff_{sca,sband,2}$

Where:

SCA, band, 0, 1, 2 are the SCA, band number, and coefficients for the updates to the Legendre polynomials. The Δ*coeff* added to the riline are the along-track updates, and the Δ*coeff* added to the risamp are the across-track updates.

rband = index to the reference band coefficient

sband = index to the search band coefficient

2.3 Calculate the new post fit Δ*errors* by updating Δ*line* and Δ*sample* with Legendre updates

Δ*line*' = Δ*line* + riline' - siline'

Δ*samp*' = Δ*samp* + risamp' - sisamp'

Where:

Δ*line* and Δ*samp* are the same as those calculated in 2.1.10 from stage 2. See note #10.

2.4 Sum post-fit variables

post-fit sum line = post-fit sum line + Δ*line*'

post-fit sum sample = post-fit sum sample + Δ*sample*'

post-fit sumsq line = post-fit sumsq line + Δ*line*'^{2}

post-fit sumsq sample = post-fit sumsq sample + Δ*sample*'^{2}

3. Calculate the post- and pre-fit statistics for the both line and sample directions:

$$\text{mean} = \frac{\text{sum}}{\text{number of points}}$$

$$\text{scale} = \frac{1}{\text{number of points} * (\text{number of points} - 1)}$$

$$\text{standard deviation} = \sqrt{\frac{\text{sum squares}}{\text{number of points} - 1} - \text{sum} * \text{sum} * \text{scale}}$$

$$\text{rmse} = \sqrt{\frac{\text{sum squares}}{\text{number of points}}}$$

Where

sum = pre/post sum line or pre/post sum sample
sum squares = pre/post sumsq line or pre/post sumsq sample
number of points = number of points used in LSQ fit

4. Create Band-to-Band Calibration output report (See Table 4-27).
   4.1 Write the report header information.
   4.2. Write the post- and pre-fit statistics (per SCA) for the line and sample direction.
   4.3. Write the individual tie-point statistics (if tie-point reporting flag = Yes).
5. If the CPF group flag is set to yes, write out the ASCII file of the CPF group with new Legendre coefficients.

4.2.10.7 Output files

The output report contains a standard header. This standard header is at the beginning of the file and contains the following:

- Date and time the file was created
- Spacecraft and instrument pertaining to measurements
- Pointing (roll) angle of the spacecraft/instrument.
- Acquisition type
- Report type (band-to-band)
- Work order ID of the process (left blank if not applicable)
- WRS path/row
- Software version that produced report.
- L0R image file name

The following items should be stored (trended) in the database with respect to the Band-to-Band Calibration algorithm:

All report header information:

Date and time
Spacecraft instrument source
Work order ID
WRS path/row
Software version

Off-nadir angle
L0Rp file name
Processing file name

The following processing parameters:

L0R product ID
Bands processed
SCAs processed

The following report file information:

Number of points used per SCA
Computed Legendre along-track coefficient updates
Computed Legendre across-track coefficient updates
New Legendre along-track coefficients (updates + existing)
New Legendre across-track coefficients (updates + existing)
Post-fit mean, standard deviation, RMSE
Pre-fit mean, standard deviation, RMSE

See note #11.

Field	Description	Trend
Date and time	Date (day of week, month, day of month, year) and time of file creation.	Yes
Spacecraft and instrument source	Landsat 8/9 and OLI/OLI-2	Yes
Processing Center	EROS Data Center SVT	No
Work order ID	Work order ID associated with processing (blank if not applicable)	Yes
WRS path/row	WRS path and row (See note #4)	Yes
Software version	Software version used to create report	Yes
Off-nadir angle	Off-nadir pointing angle of processed image file (See note #5)	Yes
Acquisition Type	Earth viewing or Lunar	Yes
L0Rp image file	L0Rp image file name used to create L1TP	Yes
Processed image file name	Name of L1TP used to create report	Yes
Number of Legendre coefficients	Number of Legendre coefficients present	Yes
Heading for pre and post fit statistics	One line of ASCII text defining pre and post statistics	
For each SCA (along and across-track directions)		
SCA number	SCA number associated with statistics	Yes
Pre fit statistics	Mean, RMSE, standard deviation, along- and across-track direction (in units of radians)	Yes
Post fit statistics	Mean, RMSE, standard deviation, along- and across-track direction (in units of radians)	Yes
For each SCA and band		
Along-track solution	Legendre along-track correction coefficients	Yes
Across-track solution	Legendre across-track correction coefficients	Yes
For each SCA and band		
Along-track updates	Updated Legendre along-track coefficients	Yes
Across-track updates	Updated Legendre across-track coefficients	Yes
For each tie-point of each SCA and Band to process	Output produced only if the tie-point results flag is set to Yes.	
Point ID	Point identifier	No
SCA number	SCA number for band combination	No

Field	Description	Trend
Reference output line	Output tie-point location in line direction	No
Reference output sample	Output tie-point location in sample direction	No
Reference input line	Reference input tie-point location in line direction	No
Reference input sample	Reference input tie-point location in sample direction	No
Search input line	Search input tie-point location in line direction	No
Search input sample	Search input tie-point location in sample direction	No
Reference band	Reference band	No
Search Band	Search band	
Measured line offset	Output space offset in line direction (from Band Accuracy Assessment residuals file)	No
Measured sample offset	Output space offset in sample direction (from Band Accuracy Assessment residuals file)	No
Pre-fit line delta	Pre-fit input space line delta/offset (Δ*line*)	No
Pre-fit sample delta	Pre-fit input space sample delta/offset (Δ*sample*)	No
Post-fit line delta	Post-fit input space line delta/offset (Δ*line*')	No
Post-fit sample delta	Post-fit input space sample delta/offset (Δ*sample*')	No

Table 4-27. Band Calibration Report File

If the CPF group creation flag is set to yes, an ASCII file containing the updated Legendre LOS should be generated. This file would contain the new Legendre LOS for each SCA for every band and would be formatted according to the CPF group that the Legendre LOS resides in. The file would also contain the file attributes CPF group with the effective dates for the LOS generated (See note #3). The SCAs and bands that were not updated should still be represented within the file; these values should be the same for post- and pre-calibration.

4.2.10.8 Notes

Some additional background assumptions and notes include the following:

1. The band calibration results currently contain the L1TP name, pre- and post-fit mean, RMSE, and standard deviation for the along- and across-track direction of each SCA, new Legendre LOS coefficients, and new CPF Legendre LOS group parameters. The individual tie-point characteristic information and (pre- and post-fit) residuals and should be added to the report file (see Table 4-27).
2. See "Using the LOS geometric resampling grid to map an output pixel location to an input pixel location" in the OLI Resampling Algroithm (4.2.4) for ols2ils functionality.
3. Table 4-28 contains the file attributes and LOS groups that should be populated with the corresponding OLI fields when the CPF group creation flag is set to yes. The CPF_Status, CPF_Name_Source, CPF_Description, and CPF_Version fields were inserted during ALIAS development by software development; these fields may or may not be present/needed for OLI processing.

Parameter Groups	Parameter Name	Data Type	Description
GROUP: LOS_LEGENDRE	Along_LOS_Legendre_B***BB_NNN***_SCA***SS***	float32 array (3 values) for each band of each SCA	Legendre polynomial coefficients defining along-track viewing angle of band number ***BB***; band name ***NNN*** and SCA ***SS*** given in radians Valid format: for each term: SN.NNNNESN, where S = "+" or "-," N = 0 to 9, and E = "E."
GROUP: LOS_LEGENDRE	Across_LOS_Legendre_B***BB_NNN***_SCA***SS***	float32 array (3 values) for each band of each SCA	Legendre polynomial coefficients defining across-track viewing angle of band number ***BB***; band name ***NNN*** and SCA ***SS*** given in radians Valid format: for each term: SN.NNNNESN, where S = "+" or "-," N = 0 to 9, and E = "E"

Table 4-28. File Attributes and LOS Groups for CPF Creation

The file name for the CPF group can follow the convention:
Legendre_coefficients_<effective begin date>_<effective end date>.odl
Where:
effective begin date = YYYYMMDD
effective end date = YYYYMMDD
YYYY = Year
MM = Month of year
DD = Day of month

4. Any kind of "non-WRS" collect besides solar or lamp; such as lunar or off-nadir viewing at the poles should have 000/000 listed as the path/row.
5. Pointing angle for lunar acquisitions would be 0.0.
6. Currently, it is not expected that any calibration will be done on anything other than the full range of Legendre coefficients (3); however, support for a range of 1-3 Legendre coefficients in the solution should remain in the system.

4.2.11 OLI Focal Plane Alignment Calibration

4.2.11.1 Background/Introduction

The OLI focal plane alignment calibration algorithm compares a precision- and terrain-corrected (L1TP) OLI panchromatic band image of a geometric calibration site with the corresponding reference image. Reference images will be constructed (offline) from mosaics of high-resolution DOQ datasets or other higher (than OLI) resolution data sources (e.g., AGRI, SPOT). Whatever the source, these high-accuracy reference datasets will be collectively referred to as "DOQ" images. Each separated-SCA L1TP image is compared to the reference to measure SCA-specific deviations from the scene-average registration. The measured deviations are used to estimate corrections to the Legendre polynomial coefficients that model the nominal panchromatic band lines-of-sight (LOSs) for each SCA.

The algorithm is implemented in two steps: 1) a mensuration/setup step in which the separated-SCA L1TP image is correlated with the reference image to measure the

within-SCA deviations, and 2) a calibration Legendre coefficient update computation step in which the measured deviations are used to compute line-of-sight model correction Legendre coefficients that adjust the original LOS model to minimize the residual image deviations. The calibration update step includes applying an outlier filter to the image measurements. Separating the algorithm into two distinct steps makes it possible to run the calibration update step multiple times, using different outlier filter thresholds, for example, without having to perform the time-consuming image mensuration/correlation setup procedure more than once.

Results from individual calibration scenes are stored in the geometric trending database so that results from multiple scenes can be analyzed together when deciding whether and how to adjust the operational focal plane calibration. When a focal plane calibration update is generated, the other spectral bands would subsequently be re-registered to the panchromatic band using the band alignment calibration procedure.

4.2.11.2 Dependencies

The OLI focal plane alignment calibration algorithm assumes that the L1TP process flow has created a substantially cloud-free SCA-separated (nadir-viewing) path-oriented L1TP panchromatic band image, over a geometric calibration site, which has been registered to a reference image by using DOQ control in the LOS model correction procedure. Note that either the L1TP image will be generated to exactly match the reference DOQ image frame, or the DOQ reference image will have to be resampled to match the L1TP as a preprocessing step. This algorithm also assumes that the CPF, precision LOS model, precision grid file, and DEM used to produce the L1TP image are available.

4.2.11.3 Inputs

The OLI focal plane alignment calibration algorithm uses the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data). The second column shows which algorithm step (image mensuration or correction model computation) uses the input.

Algorithm Inputs	Processing Step
ODL File (implementation)	Both
CPF Name	Both
L1TP Image File Name	Step 1
Precision LOS Model File Name	Step 1
Precision LOS Grid File Name	Step 1
DEM File Name	Step 1
Reference Image File Name	Step 1
Correlation Data File Name	Both
Report File Name (see note #5)	Step 2
Number of Tie Points per Cell	Step 1
Outlier tolerance	Step 2
L0Rp ID (for trending)	Step 2
Work order ID (for trending)	Step 2
WRS Path (for trending)	Step 2

Algorithm Inputs	Processing Step
WRS Row (for trending)	Step 2
Calibration effective dates for updated parameters	Step 2
Trending flag	Step 2
CPF	Both
Algorithm Parameters (formerly system table parameters)	
Size of Correlation Window (see note #5)	Step 1
Peak Fit Method	Step 1
Min Correlation Strength (see note #5)	Step 1
Max Correlation Displacement (see note #5)	Step 1
Fill Threshold Fraction (max percent of window containing fill value)	Step 1
Tie point weight (in units of 1/microradians^2)	Step 2
Fit order (see note #5)	Step 2
Post-fit RMSE Thresholds (trending metrics)	Step 2
Precision Grid File (see LOS Projection Algorithm for details)	Step 1
Number of SCAs	Step 1
For each SCA:	Step 1
Grid cell size in lines/samples	Step 1
Number of lines/samples in grid	Step 1
Number of z-planes, zero z-plane index, z-plane spacing	Step 1
Array of grid input line/sample locations	Step 1
Array of output line/sample locations (per z-plane)	Step 1
Array of forward mapping coefficients	Step 1
Array of inverse mapping coefficients	Step 1
Rough mapping polynomial coefficients	Step 1
Precision LOS Model File (see the LOS Model Creation Algorithm for details)	Step 1
OLI Pan Along-Track IFOV (in radians)	Step 1
Number of SCAs	Step 1
Number of Bands	Step 1
Number of Detectors per SCA per Band	Step 1
Focal Plane Model Parameters (Legendre Coefficients) (in radians)	Step 1
L1TP Image (separated SCA)	Step 1
Image corner coordinates	Step 1
Pixel size (in meters)	Step 1
Image size	Step 1
Search image pixel data (panchromatic)	Step 1
DEM	Step 1
DEM corner coordinates	Step 1
Pixel size (in meters)	Step 1
DEM size	Step 1
Elevation data	Step 1
Reference Image	Step 1
Image corner coordinates	Step 1
Pixel size (in meters)	Step 1
Image size	Step 1
Reference image pixel data	Step 1
Correlation Data File (output of Step 1)	Step 2
Correlation results in input space pixels	Step 2
LOS errors in radians	Step 2
Correlation results in output space pixels	Step 2

4.2.11.4 Outputs

Step 1: Focal Plane Alignment Setup
Correlation Data File (temporary output passed to Legendre step)
Correlation results in output space pixels
Correlation results in input space pixels
LOS errors in radians
Step 2: Focal Plane Alignment Legendre (see Table 4-30 below)
Report File (see Table 4-30 below for details)
Standard report header
Acquisition date
Ref (DOQ)/Search (L1TP) image names
Number of SCAs
For each SCA:
SCA Number
Old Along- and Across-track Legendre coefficients (NSCAx2x3)
Along- and Across-track Legendre error (fit) coefficients (NSCAx2x3)
New Along- and Across-track Legendre coefficients (NSCAx2x3)
Pre-fit along- and across-track residual statistics (mean, stddev, RMSE)
Post-fit along- and across-track residual statistics (mean, stddev, RMSE)
Confidence level used for outlier rejection
Legendre polynomial fit order
Number of tie points used for current SCA
CPF LOS_LEGENDRE Group (ODL format file)
Effective Dates (embedded in file name)
New Legendre polynomial coefficients (NSCAx2x3)
Measured Tie Point Data
For each point:
SCA Number
Grid Cell Column Number
Nominal Output Space Line
Nominal Output Space Sample
Measured LOS Error Delta Line (in pixels)
Measured LOS Error Delta Sample (in pixels)
Measured LOS Error Along-Track Delta Angle (in microradians)
Measured LOS Error Across-Track Delta Angle (in microradians)
Tie-Point State (outlier) Flag
Along-Track Fit Residual (in microradians)
Across-Track Fit Residual (in microradians)
Focal Plane Alignment Trending Database (see Table 4-30 below for details)
L0Rp ID
Work Order ID
WRS path/row
Acquisition date
Ref (DOQ) image name
Number of SCAs
For each SCA:
SCA Number
Old Along- and Across-track Legendre coefficients (NSCAx2x3)

Along- and Across-track Legendre error (fit) coefficients (NSCAx2x3)
New Along- and Across-track Legendre coefficients (NSCAx2x3)
Pre-fit along- and across-track residual statistics (mean, stddev, RMSE)
Post-fit along- and across-track residual statistics (mean, stddev, RMSE)
Confidence level used for outlier rejection
Number of tie points used for the current SCA

4.2.11.5 Options

Focal Plane Alignment Calibration Trending On/Off Switch

4.2.11.6 Procedure

The Focal Plane Alignment Calibration Algorithm is used for on-orbit calibration of the alignment of the LOSs of the SCAs, relative to each other. This calibration is necessary to meet the image registration, geodetic accuracy, and geometric accuracy requirements.

Procedure Overview

The focal plane alignment algorithm adjusts each SCA to a known stable reference. By aligning each SCA to a common reference, any measured inter-SCA misalignment is removed. Each SCA is correlated against a reference image created from a mosaic of DOQ images. A new set of 2nd order Legendre LOS coefficients, representing updates or corrections to the original polynomials, are generated by fitting a set of coefficients to the residuals.

Substantially cloud-free scenes should be used for focal plane alignment calibration. The imagery should have ground control applied and terrain displacements removed, i.e., the imagery should be a terrain-corrected (L1TP) dataset.

4.2.11.6.1 Stage 1: Setup – Correlate L1TP Panchromatic Image with DOQ Reference

An array of test points is generated for each SCA based on the number of points per grid cell specified in the input parameters. The OLI geometric grid is used to generate the test point array by spacing the test points at regular intervals in input space, and then computing the corresponding output space coordinates for each. Constructing the test point array in input space ensures that the test points fall within the active area of each SCA.

Image windows extracted from the L1TP image at the test point locations are correlated with corresponding windows extracted from the reference DOQ image, using normalized gray-scale correlation. This procedure is the same as that described in the OLI Image Registration Accuracy Assessment Algorithm in Section 4.2.8. Since the expected offsets are small, the L1TP and DOQ image windows are the same size. The correlation procedure yields measured deviations (or correlation failure flag) in the line

and sample directions, estimated to subpixel accuracy. These deviations, or residuals, are in units of output space pixels.

The residuals measured in output space are converted to differences in LOS along- and across-track angles by mapping the reference point location from output space to input space and then mapping the search point location from output space to input space. The mappings are performed using the OLI geometric grid that was used to resample the L1TP image, and include the test point elevation interpolated from the input DEM. The OLI Resampling Algorithm in Section 4.2.4 describes this three-dimensional output space to input space mapping (3d_ols2ils). Once an input space location is found for both points, the LOS vectors are calculated for each input sample location using the OLI LOS model. The Find LOS section of the OLI LOS Projection Algorithm (4.2.2) describes this.

The angular differences in input space are then:

$$\text{along track residual} = \frac{r_x}{r_z} - \frac{s_x}{s_z} + (\text{input reference line} - \text{input search line}) * \text{along track IFOV}$$

$$\text{across track residual} = \frac{r_y}{r_z} - \frac{s_y}{s_z}$$

where:

r_x, r_y, r_z = reference x,y,z vector components of LOS
s_x, s_y, s_z = search x,y,z vector components of LOS
input reference line = input line location for reference point
input search line = input line location for search point

4.2.11.6.2 Stage 2: Legendre – Compute Focal Plane Calibration Update

A weighted least-squares routine is used to generate the fit between the angular residuals and the updated Legendre polynomial coefficients. The weight matrix [**W**] is an NxN diagonal matrix, where the diagonal elements are a user-entered weight value.

$$[\mathbf{W}]_{NxN} = \begin{bmatrix} w & 0 & \dots & 0 & \dots & 0 \\ 0 & w & 0 & \dots & 0 & \vdots \\ \vdots & 0 & w & 0 & \dots & 0 \\ 0 & \dots & 0 & \ddots & 0 & \vdots \\ \vdots & 0 & \dots & 0 & w & 0 \\ 0 & \dots & 0 & \dots & 0 & w \end{bmatrix}$$

where w = user entered weight

The weight matrix was included in order to make it possible to differentially weight the measured deviations based on correlation strength, but this is not implemented in the baseline algorithm. Instead, a common weight, read from the CPF, is used for all points.

The observation matrix is an Nx1 matrix containing the correlation residuals in input space angular units. There are two observation matrices, one for the along-track residuals and one for the across-track residuals.

$$[\mathbf{Y}]_{Nx1} = \begin{bmatrix} residual_1 \\ residual_2 \\ \vdots \\ residual_N \end{bmatrix}$$

The design matrix is an Nx3 matrix with each row of the matrix being equal to the Legendre polynomial term associated with the reference sample location of the corresponding residual measurement. The calculation of these Legendre polynomial terms, as functions of the input sample location, is described below.

$$[X]_{Nx3} = \begin{bmatrix} l_{0,1} & l_{1,1} & l_{2,1} \\ l_{0,2} & l_{1,2} & l_{2,2} \\ l_{0,3} & l_{1,3} & l_{2,3} \\ \vdots & \vdots & \vdots \\ l_{0,N} & l_{1,N} & l_{2,N} \end{bmatrix}$$

where:

$l_{i,j}$ = i^{th} Legendre polynomial term associated with reference sample location of the j^{th} residual measurement.

The solution for the updates to the original Legendre LOS coefficients can be found from:

$$[coeff] = ([X]^T[W][X])^{-1}([X]^T[W][Y])$$

The matrix [*coeff*] is a 1x3 matrix containing the corrections to be applied to the original Legendre coefficients. A separate solution is found for the along- and across-track component. The mean offset across all the SCAs corresponds to the mean residual precision correction error for the calibration scene. It is subtracted from the coefficients generated for each SCA to avoid introducing any net pointing bias into the focal plane calibration. Once the mean is subtracted, the corrections are then added to the original Legendre LOS coefficients to compute the updated focal plane alignment parameters.

Only the PAN band is used for focal plane alignment. Band calibration uses the PAN band as the reference for all the other bands; therefore, the multispectral bands are aligned to the PAN band during band calibration. A band alignment calibration should be performed following an update to the focal plane (pan band) calibration to avoid degrading the band-to-band registration.

Figure 4-45 shows the architecture for the setup portion of the Focal Plane Alignment Calibration algorithm.

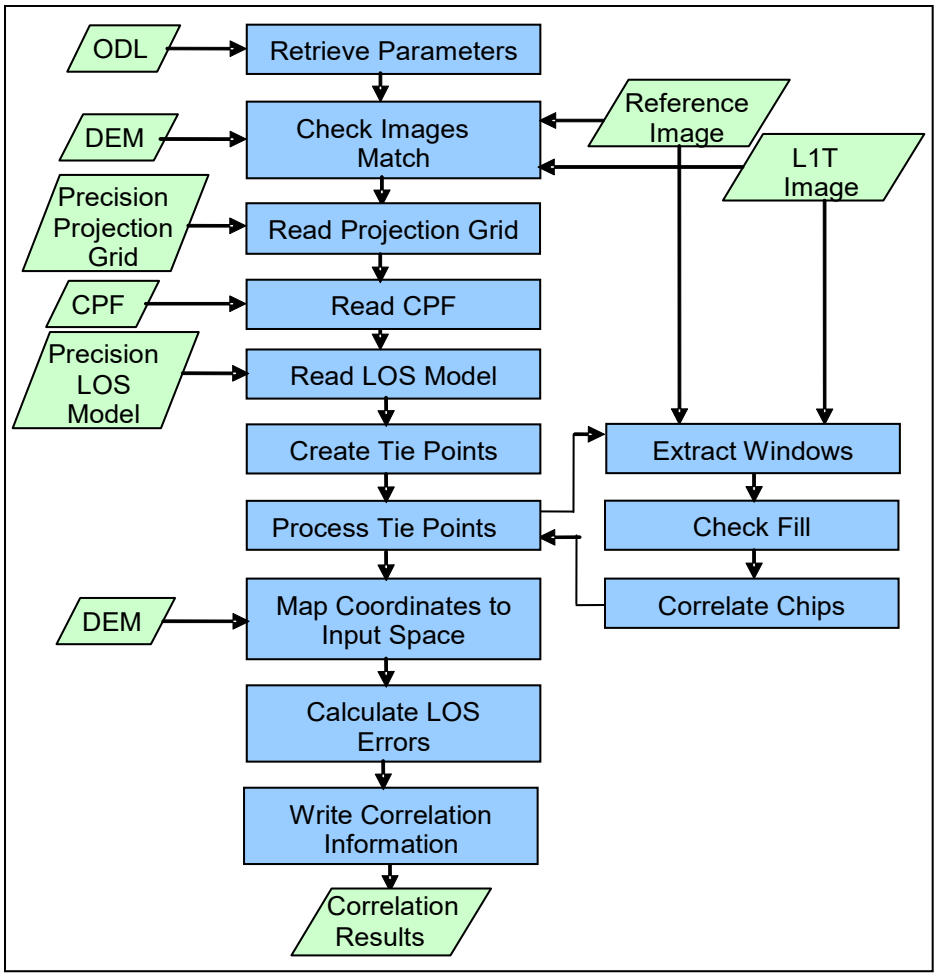

Figure 4-45. Focal Plane Calibration Setup Algorithm Architecture

Figure 4-46 shows the architecture of the Legendre polynomial fitting portion of the Focal Plane Alignment Calibration algorithm.

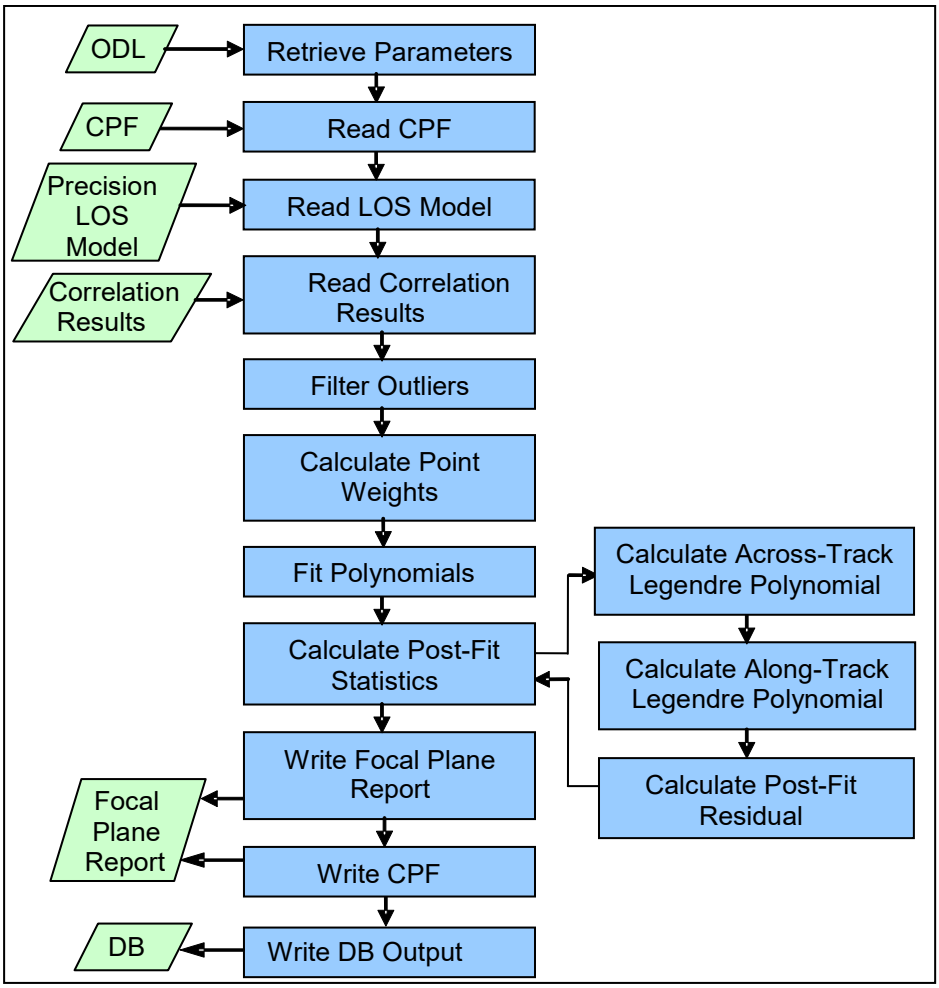

Figure 4-46. Focal Plane Calibration Legendre Polynomial Generation Algorithm Architecture

Focal Plane Cal Setup Sub-Algorithm

The setup portion consists of correlating points between the search image and the reference image, and converting the correlation offsets into LOS deviations that can be used to model each SCA's detector array, modeled by a quadratic Legendre polynomial. The results of this program are used in the second portion of focal plane calibration, the generation of new Legendre polynomials. This program creates a temporary output file that is read by the second portion of focal plane calibration.

Select Correlation Points Sub-Algorithm

To ensure evenly distributed tie point locations during correlation, locations are defined to lie at the center of each resampling grid cell, or sub-cell. There will be pts_per_cell equally-spaced points per grid cell. For example, if there are 4 points per cell, they will be placed as shown in Figure 4-47.

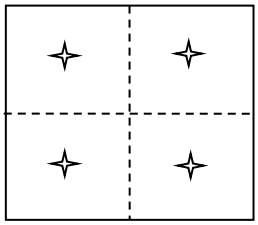

Figure 4-47. Correlation Point Placement in Grid Cell

The cell is divided into a 2-by-2 grid of 4 sub-cells, and each sub-cell is divided in half to place the point in the middle, yielding points at (0.25,0.25), (0.75,0.25), (0.25,0.75), and (0.75,0.75).

The calculation of the output space line/sample coordinates of the tie points is done as follows:

a) Compute the number of rows and columns of tie points in each cell.
   ncol = (int)ceiling(sqrt(pts_per_cell))
   nrow = (int)ceiling((double)pts_per_cell/(double)ncol)
   This creates an array of tie points containing at least pts_per_cell points.

b) For each tie point, i = 1 to ncol and j = 1 to nrow:
   b1) Compute the grid cell fractional location (cfrac,rfrac).

$$cfrac = \frac{2i-1}{2\ ncol} \qquad rfrac = \frac{2j-1}{2\ nrow}$$

   b2) Compute the output space line, ol_{ij}, using bilinear interpolation on the output line numbers at the grid cell corners, where l_{UL}, l_{UR}, l_{LL}, and l_{LR} are the output space line coordinates at the grid cell upper-left, upper-right, lower-left, and lower-right corners, respectively:

$ol_{ij} = l_{UL} * (1\text{-cfrac})*(1\text{-rfrac})$
$\quad + l_{UR} * cfrac * (1\text{-rfrac})$
$\quad + l_{LL} * (1\text{-cfrac}) * rfrac$
$\quad + l_{LR} * cfrac * rfrac$

   b3) Compute the output space sample, os_{ij}, using bilinear interpolation on the output sample numbers at the grid cell corners, where s_{UL}, s_{UR}, s_{LL}, and s_{LR} are the output space sample coordinates at the grid cell upper-left, upper-right, lower-left, and lower-right corners, respectively:

$os_{ij} = s_{UL} * (1\text{-cfrac})*(1\text{-rfrac})$
$\quad + s_{UR} * cfrac * (1\text{-rfrac})$
$\quad + s_{LL} * (1\text{-cfrac}) * rfrac$
$\quad + s_{LR} * cfrac * rfrac$

   Note that the bilinear weights are the same for the line and sample computations and only need be computed once.

The heritage version of this sub-algorithm locates the points by computing the intersection of the cell diagonals and then calculating offsets from that point. It is more straightforward to simply use bilinear interpolation, as described above, to

calculate the tie point output space coordinates, so this unit has been reworked from the heritage implementation (see note #2).

Map Coordinates to Input Space Sub-Algorithm
This function uses the inverse mapping coefficients in the grid to calculate the input space line/sample for each output space line/sample. It does this for both the reference image line/sample location and the search (L1TP) image line/sample location.

For each SCA

For each residual

1. Interpolate a height from the DEM at the location corresponding to the tie-point reference image line/sample coordinates.
2. Map the reference output line/sample location to its corresponding input line/sample location using the oli_3d_ols2ils routine.
3. Interpolate a height from the DEM at the location corresponding to the tie-point search image line/sample coordinates.
4. Map the search output line/sample location to its corresponding input line/sample location using the oli_3d_ols2ils.

Calculate LOS Errors Sub-Algorithm
This function uses the tie-point reference and search input space locations to calculate the angular line-of-sight errors.

For each SCA

For each residual

1. Calculate reference line of sight vector for sample location using the nominal detector type and the precision LOS model.
2. Calculate the search LOS vector for the sample location using the nominal detector type and the precision LOS model.
3. Calculate the residual errors in terms of the difference in the LOS along- and across-track angles:

along-track LOS error = ref los.x/los.z – srch los.x/los.z
+ input line error * along-track pan IFOV
across-track LOS error = ref los.y/los.z – srch los.y/los.z

Output Correlation Information Sub-Algorithm
The correlation points are dumped to a binary file, so the second phase of focal plane calibration (Legendre polynomial generation) can read them directly back in. First, a long integer is written to indicate the number of records, then all of the records are written. Each record contains the fields shown in Table 4-29.

Type	Field	Description
Int	sca_number	SCA number (0-relative)

Type	Field	Description
Int	grid_column	grid column number (0-relative)
Int	grid_row	grid row number (0-relative)
double	nom_os_pt.line	nominal output space point line
double	nom_os_pt.samp	nominal output space point sample
double	ref_os_pt.line	reference output space line
double	ref_os_pt.samp	reference output space sample
double	srch_os_pt.line	search output space line
double	srch_os_pt.samp	search output space sample
double	ref_is_pt.line	reference input space line
double	ref_is_pt.samp	reference input space sample
double	srch_is_pt.line	search input space line
double	srch_is_pt.samp	search input space sample
double	los_err.line	angular along-track LOS error
double	los_err.samp	angular across-track LOS error
double	los_err_pix.line	line LOS error in pixels
double	los_err_pix.samp	sample LOS error in pixels
double	correlation_accuracy	correlation accuracy
ActiveFlag	active_flag	correlation success flag
double	pt_weight	point weight for use in fit
double	fit_residual.line	line residual from fit of pts
double	fit_residual.samp	sample residual from fit of pts

Table 4-29. Correlation Record Fields

This is not a human-readable (ASCII) file, because it is only used to transport information from the first phase of calibration to the second. If the file already exists, it will be overwritten.

Generate Legendre Polynomials Sub-Algorithm
The generate Legendre portion reads the results of the focal plane setup, filters the outliers, fits the data to a Legendre polynomial, updates the SCA models, and generates output reports. This process is outlined below.

a) For each SCA

a.1) Build design matrix

Calculate normalized detector for reference sample location (see note #1). Note that in this context the “detector” number is the input sample number within the current SCA.

$$\text{normalized detector} = \frac{2 * \text{detector}}{\text{number of detectors}} - 1$$

where:
detector = reference sample location
number of detectors = number of pan band detectors in current SCA

Calculate “row” of design matrix and store in matrix

$l_{0,j}$ = 1
$l_{1,j}$ = normalized detector
$l_{2,j}$ = 1.5 *(normalized detector)^2– 0.5

where j = residual number

a.2) Build weight matrix using the (fixed) input weight value.

a.3) Build line and sample observation matrices from residuals in terms of angular differences.

a.4) Solve for line and sample solutions using weighted least-squares routine (see the Fit Polynomials sub-algorithm below).

a.5) Calculate pre-fit residual statistics from the original measured deviations.

a.5.1) Calculate statistics for along- and across-track residuals used in a.3.

Compute mean, standard deviation and RMSE for the residuals. Values are calculated for along- and across-track directions independently.

For along-track residuals:

a.5.1.1) Calculate mean
a.5.1.2) Calculate standard deviation
a.5.1.3) Calculate RMSE

For across-track residuals:

a.5.1.4) Calculate mean
a.5.1.5) Calculate standard deviation
a.5.1.6) Calculate RMSE

a.6) Calculate post fit residuals statistics for correction coefficients

Post-fit residuals are calculated by updating the original residual deviations used in step a.3 above using the Legendre polynomial corrections. Statistics are then calculated on these updated residuals.

a.6.1) For each residual

a.6.1.1) Calculate normalized detector for reference sample location (as shown in a.1)

a.6.1.2) Calculate updated along-track LOS angle from along-track Legendre coefficients calculated in a.4 (see Calculate Legendre Polynomial sub-algorithm below).

a.6.1.3) Find difference between along-track angle from a.6.1.2 and along-track angular residual

a.6.1.4) Calculate updated across-track LOS from across-track Legendre coefficients calculated in a.4

a.6.1.5) Find difference between across-track angle from a.6.1.4 and across-track angular residual

a.6.2) Calculate statistics for along and across-track updated residuals calculated in a.6.1.

Compute mean, standard deviation and RMSE for updated residuals. Values are calculated for along- and across-track directions independently.

For along-track residuals:

a.6.2.1) Calculate mean
a.6.2.2) Calculate standard deviation
a.6.2.3) Calculate RMSE

For across-track residuals:

a.6.2.4) Calculate mean
a.6.2.5) Calculate standard deviation
a.6.2.6) Calculate RMSE

b) Remove bias from the correction coefficients

sum along = sum across = 0.0

b.1) For all SCAs:

sum along = sum along + $coeff\_along_{0,sca}$

sum across = sum across + $coeff\_across_{0,sca}$

b.2) Compute the average across all SCAs:

$$\text{mean along} = \frac{\text{sum along}}{\text{number of scas}}$$

$$\text{mean across} = \frac{\text{sum across}}{\text{number of scas}}$$

b.3) For all SCAs:

$coeff\_along_{0,sca}$ = $coeff\_along_{0,sca}$ – mean along

$coeff\_across_{0,sca}$ = $coeff\_across_{0,sca}$ – mean across

c) For each SCA, add the correction coefficients to original Legendre LOS coefficients:

new along $legendre_{sca,i}$ = update along $legendre_{sca,i}$ + old along $legendre_{sca,i}$

new across $legendre_{sca,i}$ = update across $legendre_{sca,i}$ + old across $legendre_{sca,I}$

where:
i = 0,1,2 Legendre polynomial number
sca = SCA number

Filter Outliers Sub-Algorithm
This function separates the focal plane correlation data into groups for each SCA for the X (sample) and Y (line) directions. It then finds the standard deviation for the points in each group. Outlier rejection is then performed on the points based on the tolerance selected by the user and the Student's T distribution. This procedure is described in the Geometric Accuracy Assessment Algorithm in Section 4.2.6.

Calculate Point Weights Sub-Algorithm
This function calculates the weight associated with each correlation point for doing the Legendre polynomial fit.

Currently, this routine assigns the weight passed in to each point, effectively assigning each point an equal weight. Originally, it was thought that the correlation strength would factor into the weight, but that was determined to not be needed. This routine was left in to allow point-specific weight factors to be added later.

Fit Polynomials Sub-Algorithm
This function performs the weighted least-squares fit of the correlation data points (using the angular error) to find the Legendre error polynomial. The least squares correction parameter vector **X** is given by solving:

$$\mathbf{A}^T\mathbf{WAX} = \mathbf{A}^T\mathbf{WY}$$

Where:
A is the [corr_points x NUMBER_COEFFS] design matrix

W is the weight matrix including the weight for each correlation data point of the **A** matrix (only the diagonal contains weights for that row)

$\mathbf{A}^T$ being the transpose of the **A** matrix

X is the [NUMBER_COEFFS x 1] matrix with the Legendre coefficients we are looking for

Y is the [corr_points x 1] matrix of angular errors for each correlation data point corresponding to the rows in the **A** matrix

Solving the above equation for **X** yields: $\mathbf{X} = (\mathbf{A}^T\mathbf{W}\mathbf{A})^{-1} * \mathbf{A}^T\mathbf{W}\mathbf{Y}$

An optimization is used to keep the size of the matrices to a manageable level. Since the [**W**] matrix (weight) only contains data on the diagonal the $\mathbf{A}^T\mathbf{W}\mathbf{A}$ normal equation matrix can be accumulated in a [NUMBER_COEFFS x NUMBER_COEFFS] matrix and the $\mathbf{A}^T\mathbf{W}\mathbf{Y}$ matrix can be accumulated in a [NUMBER_COEFFS x 1] matrix. The $\mathbf{A}^T\mathbf{W}$ and $\mathbf{A}^T\mathbf{W}\mathbf{A}$ matrices are the same for both along- and across-track directions.

Calculate Legendre Polynomial Sub-Algorithm
This function calculates the Legendre polynomial for the input normalized detector value, x:

$along = coeff\_along_0 + coeff\_along_1 \, x + coeff\_along_2 \, (1.5*x^2 - 0.5)$

$across = coeff\_across_0 + coeff\_across_1 \, x + coeff\_across_2 \, (1.5*x^2 - 0.5)$

Write Focal Plane Calibration Results to Characterization Database
This function writes the results of the SCA Legendre polynomial fit calculations to the geometric characterization database. The output is only written to the database if the post-fit along- and across-track RMSE statistics are all below the threshold values specified in the CPF (the trending metrics). The characterization database output is listed in Table 4-30 below.

Write SCA Parameters CPF Sub-Algorithm
This function writes the updated LOS_LEGENDRE parameter group of the CPF, in the ODL format used by the CPF, to a separate output file named "LOS_LEGENDRE_yyyymmdd_YYYYMMDD.cpf," where yyyymmdd is the user-specified start effective date and YYYYMMDD is the user-specified ending effective date. Current plans call for actual calibration updates to be based on multiple scene results extracted from the characterization database, so this capability is primarily a convenience for testing purposes.

Algorithm Output Details
The contents of the output focal plane alignment calibration report file and the corresponding geometric characterization database outputs are summarized in Table 4-30 below. All fields are written to the output report file but only those with "Yes" in the "Database Output" column are written to the characterization database. Note that the first eleven fields listed constitute the standard report header.

Field	Description	Database Output
Date and time	Date (day of week, month, day of month, year) and time of file creation.	Yes
Spacecraft and instrument source	Landsat 8/9 and OLI/OLI-2	Yes

Field	Description	Database Output
Processing Center	EROS Data Center SVT (see note #4)	Yes
Work order ID	Work order ID associated with processing (blank if not applicable)	Yes
WRS path	WRS path number	Yes
WRS row	WRS row number	Yes
Software version	Software version used to create report	Yes
Off-nadir angle	Scene off-nadir roll angle (in degrees) (only nadir-viewing scenes are used for focal plane calibration)	Yes
Acquisition type	Earth, Lunar, or Stellar (only Earth-viewing scenes are used for focal plane calibration)	Yes
L0Rp ID	Input L0Rp image ID	Yes
L1T image file	Name of L1TP used to measure tie points	No
Acquisition date	Date of L1TP image acquisition (new)	Yes
Reference image file	Name of reference (DOQ) image used to measure tie points	Yes
Confidence Level	Confidence level used for outlier rejection	Yes
Fit Order	Order of Legendre fit	Yes
Number of SCAs	Number of SCAs calibrated (14) (new)	Yes
For each SCA:		
SCA Number	Number of the current SCA (1-14)	Yes
Original AT Legendre coeffs	Original along-track Legendre coefficients: a0, a1, a2	Yes
Original XT Legendre coeffs	Original across-track Legendre coefficients: b0, b1, b2	Yes
Error AT Legendre coeffs.	The computed updates to the along-track Legendre coefficients: c0, c1, c2	Yes
Error XT Legendre coeffs.	The computed updates to the across-track Legendre coefficients: d0, d1, d2	Yes
New AT Legendre coeffs	New along-track Legendre coefficients: a'0, a'1, a'2	Yes
New XT Legendre coeffs	New across-track Legendre coefficients: b'0, b'1, b'2	Yes
Pre-fit AT residual statistics	Pre-fit along-track residual mean, standard deviation, and RMSE statistics	Yes
Pre-fit XT residual statistics	Pre-fit across-track residual mean, standard deviation, and RMSE statistics	Yes
Post-fit AT residual statistics	Post-fit along-track residual mean, standard deviation, and RMSE statistics	Yes
Post-fit XT residual statistics	Post-fit across-track residual mean, standard deviation, and RMSE statistics	Yes
Number of Points	Number of tie points used for current SCA	Yes
CPF Group:		
Effective Date Begin	Beginning effective date of CPF group: YYYY-MM-DD	No
Effective Data End	Ending effective date of CPF group: YYYY-MM-DD	No
For each SCA:		
New Legendre polynomial coefficients	Nine (one per band) arrays of three along-track Legendre coefficients followed by nine arrays of three across-track Legendre coefficients.	No
Tie Point Data:	For each tie point:	
SCA Number	SCA where the tie point was measured	No
Grid Cell Column Number	Column number of the grid cell containing the tie point	No

Field	Description	Database Output
Nominal Output Space Line	Predicted tie point output space line location	No
Nominal Output Space Sample	Predicted tie point output space sample location	No
LOS Line Error	Measured LOS error delta line (in pixels)	No
LOS Sample Error	Measured LOS error delta sample (in pixels)	No
LOS AT Error	Measured LOS error along-track delta angle (in microradians)	No
LOS XT Error	Measured LOS error across-track delta angle (in microradians)	No
State Flag	Tie point state (outlier) flag	No
AT Fit Residual	Along-track fit residual (in microradians)	No
XT Fit Residual	Across-track fit residual (in microradians)	No

Table 4-30. Focal Plane Calibration Output Details

Accessing the Focal Plane Calibration Results in the Characterization Database
Though not part of the formal focal plane calibration algorithm, some comments regarding the anticipated methods of accessing and analyzing the individual scene focal plane calibration results stored in the characterization database may assist with the design of the characterization database (see note #3).

The database output from the focal plane alignment calibration algorithm will be accessed by a data extraction tool that queries the characterization database to retrieve focal plane calibration results from multiple scenes. The only processing required on the returned results is to compute the average "new" Legendre coefficients for each SCA across all returned scenes. The returned scene results and computed mean Legendre coefficient values will be output in a report containing a comma-delimited table of the retrieved trending results as well as the summary averages.

The geometric results would typically be queried by acquisition date and/or WRS path/row. The most common query would be based on acquisition date range, for example, selecting all of the results for a given calendar quarter:

Acquisition_Date is between 01APR2012 and 30JUN2012

The average coefficients would be calculated from the "new" Legendre coefficients for the individual scenes returned, as:

$$Coeff_{SCA,j,net} = \frac{1}{numScene} \sum_{i=1}^{numScene} Coeff_{SCA,j,i}$$

for coefficient j (j=0,1,2) for each SCA.

The query results would be formatted in a set of comma-delimited records (for ease of ingest into Microsoft Excel), one record per scene. Each record would contain all of the "header" fields written to the characterization database (items with "Yes" in the

rightmost column of Table 4-30 above) but only the "new" Legendre coefficients for each SCA. The other fields would be retrieved using general purpose database access tools, when desired. A header row containing the field names should precede the database records.

Following the scene records the average Legendre coefficients should be written out in the same CPF/ODL syntax used in the report file. This output uses the same structure shown in the final row in Table 4-30 above, but contains the average, rather than a single scene's, Legendre coefficients.

4.2.11.7 Notes

Some additional background assumptions and notes include the following:

1. The heritage fit_polynomials procedure uses the input space reference image point locations to derive the correction polynomial coefficients. This has been changed to use the input space search image point locations (which will be very close) instead.
2. The heritage tie point selection sub-algorithm (select_corr_pts) is unnecessarily complicated. It has been reworked to use a simpler bilinear interpolation approach.
3. The trending output from this algorithm will be accessed by an analysis tool that queries the trending database to retrieve focal plane alignment results from multiple scenes. Averaging the Legendre coefficients derived from calibration scenes within a user-specified date range will smooth out residual precision correction errors, leading to a more consistent focal plane calibration solution. The analysis tool will create a report file containing a comma-delimited table of the retrieved trending results and the averaged Legendre coefficients.
4. A configuration table (system table) should be provided for each installation of the algorithm implementation to convey site-specific information such as the processing center name (used in the standard report header), the number of processors available (for parallel processing implementations), etc. This takes the place of the heritage system table, which also contained certain algorithm-related parameters. Anything related to the algorithms has been moved to the CPF for Landsat 8/9. For the prototype implementation, the site-specific report header fields are provided as environment variables.
5. Heritage optional input parameters that allow the report output to be suppressed and that over-ride several of the processing parameters now provided in the CPF (minimum correlation strength, maximum correlation offset, correlation window size, and Legendre fit order) have been retained in the prototype implementation to facilitate testing, but they are not required.

4.2.12 OLI Sensor Alignment Calibration Algorithm

4.2.12.1 Background/Introduction

The OLI sensor alignment calibration algorithm uses a time sequence of the LOS model alignment trending results generated by the LOS model correction algorithm (see the LOS Model Correction Algorithm in Section 4.2.3 for details) to estimate the orientation of the OLI coordinate system relative to the spacecraft attitude determination coordinate system. This spacecraft -to-OLI alignment is one of the fundamental geometric calibration parameters stored in the CPF. Analyzing time sequences of measured alignment results makes it possible to smooth out random scene-to-scene pointing errors to estimate, and correct for, any underlying systematic alignment errors. The OLI sensor alignment calibration algorithm is inspired by the ALI sensor alignment algorithm used in ALIAS. Its implementation will be different, in that the ALIAS code was set up to operate on individual scene results. The heritage logic takes the precision solution output, converts it to apparent alignment errors, and then blends the individual scene results with the current best estimate of the alignment state using a Kalman filter. This approach required the scenes to be processed in time order and did not provide a view of how the apparent alignment errors varied with time, which would have made it easier to detect systematic (e.g., seasonal) effects. By retrieving and analyzing groups of individual alignment results, the OLI algorithm will make it possible to select an appropriate time window and take all the data from that window into account when deriving the alignment calibration for that time.

4.2.12.2 Dependencies

The OLI sensor alignment calibration algorithm assumes that the LOS model correction algorithm (6.2.3) has populated the geometric trending database with LOS model alignment trending results.

4.2.12.3 Inputs

The OLI sensor alignment calibration algorithm uses the inputs listed in the following table. The user inputs define the parameters of a query used to retrieve the desired alignment trending data created by the LOS model correction algorithm.

Algorithm Inputs
LOS Model Correction Alignment Trending Data (from trending DB)
Precision correction reference date/time (year, day of year, hours, minutes, seconds)
Roll-pitch-yaw alignment angles (in microradians)
Ephemeris position corrections (in meters)
Alignment covariance matrix
Across- and along-track RMS GCP fit solution quality metrics (in meters)
Control type used (GLS or DOQ)
Number of control points used
GCP outlier threshold used
GCP RMS fit (in meters)
Off-nadir angle (in degrees)
Geometric characterization ID (of trended scene)
Work Order ID (of trended scene)

Algorithm Inputs
WRS Path/Row
User Inputs
Trending Data Query Date Range
Calibration Effective Date Range
Control Type Selection (GLS, DOQ, Both)
Control Type Weights (if Both are used) (see note 2)
Maximum off-nadir angle (in degrees) (see note 5)
Alignment Trending Flag (1 = save results)
Calibration Parameter File Name
Output Report File Name

4.2.12.4 Outputs

OLI Alignment Report (see Table 4-31 for details)
Standard report header fields
Control type/GCP source selected
Number of scenes analyzed
Retrieved data date range
Estimated alignment angles (roll, pitch, yaw)
Measured alignment angle RMS residuals (roll, pitch, yaw)
OLI Alignment Calibration Parameters
Alignment effective date range
ACS to OLI rotation matrix
Table of alignment trending results returned
OLI Alignment Characterization Database Output
Processing date
Processing site
Maximum off-nadir angle used
Control type/GCP source selected
DOQ vs. GLS scene weights used
Number of scenes analyzed
Retrieved data date range
Estimated alignment angles (roll, pitch, yaw)
Measured alignment angle RMS residuals (roll, pitch, yaw)
Alignment effective date range
ACS to OLI alignment matrix

4.2.12.5 Options

Control Source Selection (GLS or DOQ or Both)
Alignment Calibration Trending On/Off Switch

4.2.12.6 Procedure

The purpose of the sensor alignment algorithm is to use a sequence of LOS model correction solutions, including both correction parameter estimates and estimated covariance information, to estimate the underlying ACS frame to OLI instrument frame alignment. A weighted least-squares batch filter implementation is used to isolate the systematic alignment trend from the scene-to-scene variability of the attitude and ephemeris precision correction errors.

Unlike the other geometric correction, characterization, and calibration algorithms, the operational implementation of this algorithm will rely upon an interactive user interface

that queries the geometric characterization database to retrieve a user-specified (based on date range and/or control source) set of LOS model correction alignment characterization results. The prototype implementation emulates this process by providing query parameters in an ODL parameter file and using them to filter (query) a static ASCII text file that emulates the trending database.

The algorithm uses the individual scene results returned from the database to estimate updates to the OLI alignment angles. The LOS model correction algorithm generates apparent OLI-to-ACS alignment angles each time it runs, whether on L1T product scenes using Global Land Survey (GLS) 2000 control, or on calibration scenes using digital orthophoto quadrangle (DOQ) control, based on the attitude corrections it estimates from the ground control measurements. The sensor alignment calibration algorithm analyzes these results over multiple scenes to detect the systematic trends that are used to update the ACS-to-OLI alignment estimate used in the CPF.

The sensor alignment calibration algorithm consists of five steps:

1. Allow the user to specify the date range, control source(s), and maximum off-nadir angle defining the desired range of LOS model correction alignment results.
2. Query the geometric characterization database to retrieve results meeting the specified criteria.
3. Determine the best-fit alignment angles from the individual scene results using a least squares procedure.
4. Allow the user to review the results, edit the list of input scenes used and rerun the solution, and accept or reject the result.
5. If the result is accepted by the user, generate an output report including the list of input scenes used and the final best-fit alignment parameters, compute the corresponding ACS to OLI alignment matrix, and write out a calibration parameter group containing the alignment matrix in the format used by the CPF.

The sensor alignment calibration algorithm procedure is depicted in Figure 4-48.

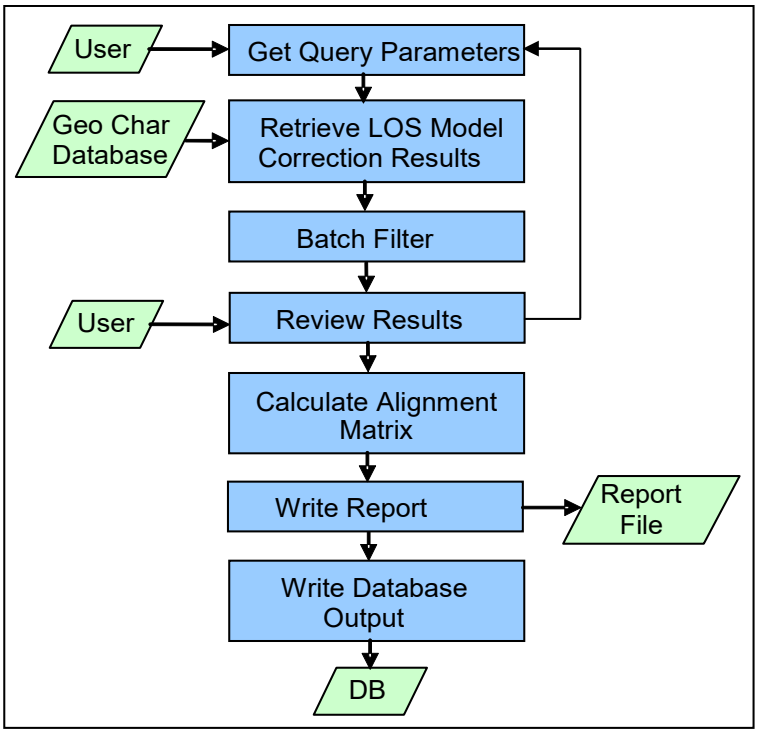

Figure 4-48. Sensor Alignment Calibration Algorithm Architecture

4.2.12.6.1 Step 1: Define the Data

The user provides a start and stop date to define the desired range of acquisition dates for the returned characterization data, a control type selection that makes it possible to use scenes processed with either DOQ or GLS control, or both, and a maximum off-nadir angle, in degrees, to include or exclude off-nadir acquisitions from the calibration process. The start/stop dates are inclusive. If the start date is not provided, all data acquired on or before the stop date are used. If the stop date is not provided, all data acquired on or after the start date are used. If no dates are provided, all dates are included. The DOQ/GLS/Both control selection defaults to DOQ. The maximum off-nadir angle is provided as an absolute value, i.e., only scenes between +MAXANG and -MAXANG would be included if MAXANG is the specified limit. The off-nadir angle limit defaults to 0.1 degrees to exclude off-nadir images.

4.2.12.6.2 Step 2: Retrieve the Data

The date range and control selection defined in step 1 are used to construct a database query to retrieve the desired scene records from the LOS model correction alignment table in the characterization database (see Table 4-31 in the LOS Model Correction Algorithm (see 4.2.3)). All fields in this table are returned and all but the full alignment covariance matrix are displayed to the user (in step 4 below). Only the diagonal elements and the roll-Y and pitch-X elements of the covariance are displayed. The fields returned include the following:

1. Work order ID,

2. Geometric Characterization ID,
3. WRS path,
4. WRS row,
5. Control type,
6. Off-nadir angle,
7. Number of GCPs used,
8. GCP outlier threshold used,
9. Root-mean-square (RMS) ground control point (GCP) fit (solution quality metric),
10. Acquisition date (year, day of year) and time (hours, minutes, seconds),
11. Measured roll alignment,
12. Measured pitch alignment,
13. Measured yaw alignment,
14. Measured ephemeris (orbital) X correction,
15. Measured ephemeris (orbital) Y correction,
16. Measured ephemeris (orbital) Z correction,
17. Alignment covariance matrix:

$Cov_{roll-roll}$	$Cov_{roll-pitch}$	$Cov_{roll-yaw}$	Cov_{roll-X}	Cov_{roll-Y}	Cov_{roll-Z}
$Cov_{pitch-roll}$	$Cov_{pitch-pitch}$	$Cov_{pitch-yaw}$	$Cov_{pitch-X}$	$Cov_{pitch-Y}$	$Cov_{pitch-Z}$
$Cov_{yaw-roll}$	$Cov_{yaw-pitch}$	$Cov_{yaw-yaw}$	Cov_{yaw-X}	Cov_{yaw-Y}	Cov_{yaw-Z}
Cov_{X-roll}	$Cov_{X-pitch}$	Cov_{X-yaw}	Cov_{X-X}	Cov_{X-Y}	Cov_{X-Z}
Cov_{Y-roll}	$Cov_{Y-pitch}$	Cov_{Y-yaw}	Cov_{Y-X}	Cov_{Y-Y}	Cov_{Y-Z}
Cov_{Z-roll}	$Cov_{Z-pitch}$	Cov_{Z-yaw}	Cov_{Z-X}	Cov_{Z-Y}	Cov_{Z-Z}

Note that the covariance matrix can be subdivided into four 3x3 blocks:

A	**B**
$\mathbf{B^T}$	**C**

Where: **A** is the covariance of the attitude correction parameters, **C** is the covariance of the ephemeris correction parameters, **B** is the cross-covariance of the attitude and ephemeris parameters, and $\mathbf{B^T}$ is the transpose of the cross-covariance.

This formulation of the covariance matrix will be used below in combining the measured alignment and ephemeris corrections.

4.2.12.6.3 Step 3: Compute the Alignment

Computing the least squares estimate of the underlying alignment trends from the retrieved sequence of individual scene alignment measurements is complicated by the correlation between the measured angular alignment corrections and the measured ephemeris corrections. Although we do not expect to detect any systematic offset in the position bias terms (x, y, and z), they are included because of their high correlation with the attitude biases. This is reflected in the observation covariance matrix where significant off-diagonal terms will exist for X/pitch and Y/roll. Any particular LOS model correction solution will resolve the correlation between the parameters by allocating the along-track and across-track errors between the ephemeris and attitude parameters based on their a priori weights. Thus, some of the systematic alignment bias could end up allocated to the ephemeris correction terms. Over multiple precision correction

solutions, the net ephemeris bias should be very close to zero. Therefore, we use the covariance information to combine the ephemeris terms with the alignment terms to create consolidated along- and across-track corrections. In practice, since accurate GPS-derived ephemeris will be available, most of the correction will be allocated to the attitude terms in the LOS model correction solutions anyway.

Each retrieved scene provides a vector of six correction measurements:

$$[X] = \begin{bmatrix} roll \\ pitch \\ yaw \\ X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix}$$

where:

roll = roll alignment angle, in microradians (μrad);
pitch = pitch alignment angle (μrad)
yaw = yaw alignment angle (μrad)
X = along-track orbit position error in meters (m)
Y = cross-track orbit position error (m)
Z = radial orbit position error (m)
x = the roll-pitch-yaw 3x1 sub-vector
y = the X-Y-Z 3x1 sub-vector

The corresponding covariance matrix is also retrieved (see the LOS Model Correction Algorithm (see Section 4.2.3) for a description of how these characterization data are created). It has the following structure:

$$CovX = \begin{bmatrix} Cov_{roll-roll} & Cov_{roll-pitch} & Cov_{roll-yaw} & Cov_{roll-X} & Cov_{roll-Y} & Cov_{roll-Z} \\ Cov_{pitch-roll} & Cov_{pitch-pitch} & Cov_{pitch-yaw} & Cov_{pitch-X} & Cov_{pitch-Y} & Cov_{pitch-Z} \\ Cov_{yaw-roll} & Cov_{yaw-pitch} & Cov_{yaw-yaw} & Cov_{yaw-X} & Cov_{yaw-Y} & Cov_{yaw-Z} \\ Cov_{X-roll} & Cov_{X-pitch} & Cov_{X-yaw} & Cov_{X-X} & Cov_{X-Y} & Cov_{X-Z} \\ Cov_{Y-roll} & Cov_{Y-pitch} & Cov_{Y-yaw} & Cov_{Y-X} & Cov_{Y-Y} & Cov_{Y-Z} \\ Cov_{Z-roll} & Cov_{Z-pitch} & Cov_{Z-yaw} & Cov_{Z-X} & Cov_{Z-Y} & Cov_{Z-Z} \end{bmatrix}$$

Comparing this to the A-B-C decomposition shown above, note that the "**A**" portion of the covariance matrix contains the attitude/attitude terms, the "**B**" portion of the matrix contains the attitude/ephemeris terms, and the "**C**" portion of the matrix contains the ephemeris/ephemeris terms.

The alignment and ephemeris corrections are combined as follows:

$$\mathbf{x}' = \mathbf{x} - \mathbf{B}\mathbf{C}^{-1}\mathbf{y}$$

where: **x'** = the consolidated along- and across-track alignment vector
x = the input alignment corrections
y = the input ephemeris corrections
B and **C** are the 3x3 covariance sub-matrices defined above.

Thus, the six LOS model correction measurements retrieved for each scene are reduced to three equivalent alignment angle observations for each scene. Consolidating the results for each scene in this manner yields a sequence of alignment angle observations:

$\mathbf{x'}_j$ where: j = 1 to N with N being the number of scenes retrieved.

Each scene observation is also assigned a (scalar) weight, w_j, based upon its control source. The DOQ and GLS weights are editable by the user, and are initially populated with default values (e.g., 50% for DOQ scenes and 50% for GLS scenes). Note that these weights are only relevant if both GLS and DOQ controlled scenes are used at the same time. The logic that enables the user to edit the weights should ensure that only numbers between 0 and 100% are allowed.

The new alignment estimate is the weighted average of these observations:

$$\theta = \frac{\sum_{j=1}^{N} w_j x'_j}{\sum_{j=1}^{N} w_j} = \begin{bmatrix} \theta_r \\ \theta_p \\ \theta_y \end{bmatrix}$$

Compute the RMS residuals:

$$RMS_r = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(roll'_j - \theta_r)^2}$$

$$RMS_p = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(pitch'_j - \theta_p)^2}$$

$$RMS_y = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(yaw'_j - \theta_y)^2}$$

The corresponding orientation matrix is:

$$M_{OLI2ACS} =$$

$$\begin{bmatrix} \cos(\theta_p)\cos(\theta_y) & \sin(\theta_r)\sin(\theta_p)\cos(\theta_y)+\cos(\theta_r)\sin(\theta_y) & \sin(\theta_r)\sin(\theta_y)-\cos(\theta_r)\sin(\theta_p)\cos(\theta_y) \\ -\cos(\theta_p)\sin(\theta_y) & \cos(\theta_r)\cos(\theta_y)-\sin(\theta_r)\sin(\theta_p)\sin(\theta_y) & \cos(\theta_r)\sin(\theta_p)\sin(\theta_y)+\sin(\theta_r)\cos(\theta_y) \\ \sin(\theta_p) & -\sin(\theta_r)\cos(\theta_p) & \cos(\theta_r)\cos(\theta_p) \end{bmatrix}$$

Take the transpose of the $\mathbf{M}_{OLI2ACS}$ matrix to compute the ACS to OLI alignment matrix used in the CPF, $\mathbf{M}_{ACS2OLI}$.

4.2.12.6.4 Step 4: Review the Results

The user is presented with a scrollable table of the individual scene results as well as the summary roll, pitch, and yaw alignment values (θ_r, θ_p, and θ_y) and the roll-pitch-yaw RMS residuals. The scene results table's columns include, in addition to the fields identified in step 2 above, the weight value, and the consolidated roll'-pitch'-yaw' values computed in step 3. The Landsat 7 Bumper Mode User Interface (BUI) is a good model for this display.

Each row in the table includes a check box or button (e.g., the BUI "Select" column) that the user can select to remove that scene/row from the alignment calculation. The user may choose to exclude scenes, based on the off-nadir angle or RMS GCP fit metrics, for example. The user can also press a button to recalculate the average alignment based on the current selection of rows. The user should be able to adjust the selected scene list and recompute the average alignment as many times as desired.

A capability to plot each alignment angle is desirable but not required (see note #1). If provided it should allow the user to select which axis (roll, pitch, or yaw) to plot and then plot the corresponding consolidated angles for each selected scene on the Y axis with scene acquisition date on the X axis. It should also show the mean alignment value for that axis as a solid horizontal line or other easily identifiable symbol.

Once the user is satisfied with the alignment solution, or is ready to give up, he or she presses either the "Accept" button or the "Quit" button. The "Accept" button advances the process to step 5. The "Quit" button terminates the algorithm.

4.2.12.6.5 Step 5: Generate the Output

The sensor alignment calibration algorithm creates either two or three outputs depending upon the setting of the "Alignment Trending Flag.” In all cases, a report file is generated using the input file name specified by the user, and an ODL file fragment is written out containing the ATTITUDE_PARAMETERS CPF group including the newly calculated ACS-to-OLI rotation matrix. Characterization database output is only created if the alignment trending flag is set to "Yes.” The user input effective date ranges for the output calibration parameters (the ACS-to-OLI sensor alignment matrix) are embedded in the automatically generated file name used for the output ODL CPF fragment. The calibration parameter effective date range need not match the original query date range as it is often desirable to include extra data from outside the calibration time window to ensure continuity in the calibration parameters from time to time period.

Once the solution is accepted by the user a report file is generated containing the items shown in Table 4-31. Note that the first 11 items in Table 4-31 constitute the standard report header, but that several of these fields are not applicable for a multi-scene algorithm such as sensor alignment calibration. Also note that the alignment matrix output is formatted as a CPF parameter group and includes the effective date range

specified by the user. In addition to the standard report header information, the report file contains the summary alignment angles and RMS residuals, the CPF OLI alignment matrix and effective dates, and a comma-delimited table containing key fields from all the individual scene rows used in the alignment solution.

If the alignment trending flag is set, the subset of the items in Table 4-31 with "Yes" in the "Database Output" column are written to the characterization database.

Field	Description	Database Output
Date and time	Date (day of week, month, day of month, year) and time of file creation.	Yes
Spacecraft and instrument source	Landsat 8/9 and OLI/OLI-2	No
Processing Center	EROS Data Center SVT (see notes #3)	Yes (see note #4)
Work order ID	Work order ID – not used for sensor alignment calibration as it operates on the results of multiple work orders.	No
WRS path	WRS path number - blank for sensor alignment cal	No
WRS row	WRS row number - blank for sensor alignment cal	No
Software version	Software version used to create report	No
Off-nadir angle	Actual maximum scene off-nadir roll angle (in degrees)	Yes
Acquisition Type	Earth viewing, Lunar, or Stellar (only Earth-viewing scenes are used for sensor alignment calibration)	No
Geo char ID	Geometric characterization trending ID – not used for sensor alignment calibration.	No
L1G image file	Not used for sensor alignment calibration	No
Acquisition date	N/A for sensor alignment calibration	No
GCP source	Ground control source used (GLS or DOQ or Both)	Yes
DOQ Weight	Weight placed on DOQ-controlled scenes (0-100%)	Yes
GLS Weight	Weight placed on GLS-controlled scenes (0-100%)	Yes
Number of scenes used	Number of scenes used in calibration	Yes
Data start date	Start date of data window used (query start)	Yes
Data stop date	Stop date of data window used (query stop)	Yes
Roll alignment	Best-fit roll alignment angle in microradians	Yes
Pitch alignment	Best-fit pitch alignment angle in microradians	Yes
Yaw alignment	Best-fit yaw alignment angle in microradians	Yes
Roll residual RMSE	RMSE of individual scene roll residuals (microradians)	Yes
Pitch residual RMSE	RMSE of individual scene pitch residuals (microradians)	Yes
Yaw residual RMSE	RMSE of individual scene yaw residuals (microradians)	Yes
Alignment effective date start	Start effective date of alignment calibration Report format: Effective_Date_Begin = "YYYY-MM-DD"	Yes
Alignment effective date stop	Stop effective date of alignment calibration Report format: Effective_Date_End = "YYYY-MM-DD"	Yes
OLI sensor alignment matrix	Best-fit ACS to OLI rotation matrix Report format: Attitude_To_OLI_Matrix =	Yes

Field	Description	Database Output
	(sn.nnnnnnnnEsnn, sn.nnnnnnnnEsnn, sn.nnnnnnnnEsnn, sn.nnnnnnnnEsnn, sn.nnnnnnnnEsnn, sn.nnnnnnnnEsnn, sn.nnnnnnnnEsnn, sn.nnnnnnnnEsnn, sn.nnnnnnnnEsnn) where: s = sign (+/-) and n = digit	
For each scene used in the calibration:		
Work order ID	Work order ID that generated scene results	No
Geo Char ID	Geometric Characterization ID for scene	No
WRS path	Scene WRS path number	No
WRS row	Scene WRS row number	No
Control type	DOQ or GLS	No
RMS GCP fit	Root-mean-square (RMS) ground control point (GCP) fit solution quality metric	No
Acquisition date	Scene acquisition date	No
Combined roll alignment	Consolidated roll value (roll') in microradians	No
Combined pitch alignment	Consolidated pitch value (pitch') in microradians	No
Combined yaw alignment	Consolidated yaw value (yaw') in microradians	No

Table 4-31. Sensor Alignment Calibration Output Details

Accessing the Sensor Alignment Characterization Database

Though not part of the formal sensor alignment calibration algorithm, some comments regarding the anticipated methods of accessing and analyzing the sensor alignment results stored in the characterization database may assist with the design of the characterization database.

The database output from the sensor alignment calibration algorithm will be accessed only for purposes of reviewing the history of calibration operations. Unlike other calibration and characterization algorithms, no summary statistics are required since the sensor alignment calibration results are themselves summary statistics. Hence, a special tool to perform the query and retrieval is probably not necessary so long as basic database query capabilities are readily available.

The sensor alignment results would typically be queried by processing date, CPF effective dates, maximum off-nadir angle, and/or GCP source. The most common query would likely be a combination of GCP source and CPF effective date range, for example, selecting all of the DOQ-derived results effective for a given calendar year:

GCP_Source = "DOQ"
Effective_Start_Date is between 01JAN2013 and 31DEC2013
Effective_Stop_Date is between 01JAN2013 and 31DEC2013

The query results should be formatted in a set of comma-delimited records (for ease of ingest into Microsoft Excel), one record per scene. Each record would contain all of the

fields written to the characterization database (items with "Yes" in the rightmost column of Table 4-31 above). A header row containing the field names should precede the database records.

4.2.12.7 Notes

Some additional background assumptions and notes include the following:

1. Plotting capability that shows the individual scene alignment measurements along with the fitted constant results would also be nice.
2. Results derived from GLS control, if used at all for sensor alignment calibration, would be given lower weight than DOQ controlled scenes due to the poorer accuracy of the GLS control source as well as the additional alignment uncertainty introduced by using the SWIR1 band for GCP mensuration.
3. Configuration parameters should be provided for each installation of the algorithm implementation to convey site-specific information such as the processing center name (used in the standard report header), the number of processors available (for parallel processing implementations), etc. This takes the place of the heritage system table, which also contained certain algorithm-related parameters. Anything related to the algorithms has been moved to the CPF for Landsat 8/9. This is implemented through environment variables in the prototype.
4. Most geometric characterization trending output is performed on a scene-by-scene basis so the work order ID, geometric characterization ID, and/or acquisition date fields uniquely identify the data being characterized and provide the basis for a unique database key for the trended record. In the case of sensor alignment, many input datasets are used, so provisions for a unique database record key are somewhat different. This is primarily an implementation detail, so no unique key fields are called out in the output table or Table 4-31 above, nor are sensor alignment calibration work order or geometric characterization IDs included in the database output. That said, the concept is that a unique sequence number would be automatically generated each time a new sensor alignment calibration record was inserted into the characterization database. In conjunction with a processing site identifier (e.g., EROS, GSFC, BATC, or SDSU), this sequence number would provide the basis for a primary key that would uniquely distinguish OLI sensor alignment calibration results generated at EROS or elsewhere.
5. Sensor alignment calibration would typically be run using the results from nadir-viewing scenes only. Since all processed scenes can produce alignment calibration results (through the LOS model correction algorithm) the user of this application needs to be able to filter the scene results based upon the off-nadir angle field.
6. Additional infrastructure not addressed in this algorithm description will be required to implement this algorithm operationally. This includes:
   a. A user interface to capture input from the Cal Analyst will be required, but is not provided with the prototype implementation.

b. An interface to query the geometric trending database will be required. This is emulated by a text file in the prototype implementation.
c. A capability to display the query results (as a table) to the Analyst, allowing him/her to selectively include or exclude particular entries will be required. This is not provided in the prototype implementation.
d. The updated alignment estimate computation capabilities perform the following:
    1) Merge the X and Y precision solution offsets into the pitch and roll (respectively) alignment estimates using the trended covariance information. Note that this blending was implicit in the heritage Kalman filter implementation.
    2) Fit constant (average) functions to the alignment estimates.
    3) Convert the alignment angles to the equivalent ACS-to-OLI rotation matrix for inclusion in the CPF.
e. A capability to insert the resulting alignment calibration parameters into the trending database (upon Analyst command) will also be required. This is emulated by ASCII output to stdout in the prototype implementation.

4.2.13 OLI MTF Bridge Characterization

4.2.13.1 Background/Introduction

The OLI Modulation Transfer Function (MTF) Characterization algorithm measures the on orbit spatial performance of the OLI using terrestrial linear targets, mostly long bridges. Historically, the Lake Pontchartrain Causeway and nearby I-10 bridge located in Louisiana have been used for this purpose since the launch of Landsat 7 in 1999. First, the Lake Pontchartrain Causeway was used to perform the across-track characterization. Then the cross sectional response to the I-10 bridge was characterized and the previously determined across-track component was backed out of the characterization, leaving the along-track component. Both bridges cross over a large area of water providing for a fairly uniform background against a well-defined step function.

The current algorithm generalizes the cross-sectional response approach used for the I-10 bridge to accommodate a wider variety of target geometries and locations. This makes a larger target list available for use, providing more spatial performance observations over a wider variety of acquisition conditions, hopefully leading to more robust estimates of OLI spatial performance. The same algorithm can be used for both the Landsat 8 OLI and the Landsat 9 OLI-2, but the formal reporting of key performance requirement (KPR) results for L8/OLI is not necessary for L9/OLI-2.

The MTF bridge characterization is accomplished in four steps. The first step extracts image pixels over the bridge targets using a pre-specified UTM region of interest to identify where pixels are, and are not, to be extracted. The DN value and UTM location of each pixel are recorded so that the pixel's distance from the bridge target centerline can be calculated. The second step creates a "super-resolution" profile of the bridge by interleaving pixels, sorted by distance from target, into nominally 1/8 pixel-sized bins, to create one profile sampled at 1/8 the original sample distance. The third step uses a

simulated annealing method to minimize the difference between a simulated target response and the “super-resolution” data to derive OLI system transfer function (STF) parameter value estimates for the direction defined by the target’s cross section. The fourth, and final, step uses target fit results from multiple cross sectional angles to estimate the underlying along- and across-track system transfer function components.

For each OLI spectral band, the characterization estimates the STF, constructs the corresponding point spread function (PSF), and edge spread function (ESF), and measures the slope of the ESF between the 40% and 60% response points. Additional edge response parameters can be derived from the constructed ESF. As noted above, the STF is estimated by comparing a simulated modeled target to actual image data. The instrument model consists of optical, detector, phase, and ground sample distance (GSD) components. The along-track model also contains an integration time component.

4.2.13.2 Dependencies

The OLI MTF bridge characterization algorithm assumes that a cloud free L1TP image has been generated and the corresponding geometric model and grid are available. The image must be in a North-up UTM projection.

4.2.13.3 Inputs

The MTF Bridge Characterization algorithm and its component sub-algorithms use the inputs listed in the following tables. Note that some of the “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data). MTF Bridge characterization does not normally include the OLI cirrus band.

4.2.13.3.1 MTF Extract Inputs

Algorithm Inputs	Contents
ODL Parameter File	Processing parameters
L0R File Name	Original L0R data (including metadata)
L1R File Name	Unresampled image values
Grid File Name	Maps output locations to input image space
OLI LOS Model File Name	Used to compute target range and velocity
Calibration Parameter File Name	Sensor and Earth constants
Band List	Bands to process
Target Definition File	ODL file containing bridge target parameters
Band List	Bands included in target definition
Number of Targets	Number of targets in current WRS
For each target:	Parameters provided for each target
Target Name	Identifying text name of bridge target
Water Reflectance (per band)	Water background signal (DN) level per band
Water Asymmetry (per band)	Asymmetry in water signal per band
Target Centerline UTM Coordinates	UTM X,Y defining start and end of target centerline
Number of Target Spans	Number of spans (pulses) in target
For each span:	Parameters provided for each span
Bridge Reflectance (per band)	Bridge signal (DN) level per band
Span Width (meters)	Bridge span width in meters

Algorithm Inputs	Contents
Span Offset (meters)	Offset (in meters) from the target centerline
Target Region of Interest (ROI)	UTM polygon(s) defining target bounds
Number of Polygons	Number of separate polygons in ROI
For each polygon:	Parameters provided for each polygon
Number of Vertices	Number of coordinates in current polygon
Coordinate List	UTM X,Y coordinates for each vertex

4.2.13.3.2 MTF Profile Inputs

Algorithm Inputs	Algorithms/sub algorithms
ODL Parameter File	Processing parameters
L0R File Name	Original L0R data (including metadata)
Band List	Bands to process
SCA List	SCAs to include (only a few will actually contain data)
Number of Profile Samples (default = 128)	Length of profile to construct
Oversampling Factor (default = 8)	Oversample factor for constructed profile
Outlier Threshold	T-Distribution Test Threshold (0.0 to 1.0)
Target Definition File Name	See MTF Extract for target definition file contents.
Pixel File Directory Name	Directory containing the MTF Extract pixel table files

4.2.13.3.3 MTF Estimate Inputs

Algorithm Inputs	Algorithm/sub-algorithms
ODL Parameter File	Processing parameters
L0R File Name	Original L0R data (including metadata)
Target Definition File Name	See MTF Extract for target definition file contents.
Profile ODL File Directory Name	Directory containing the MTF Profile output ODL files
Band List	Bands to process
Solution Method Switch	Selects Numerical Recipes (heritage) implementation (0) or GSL implementation (1) of simulated annealing
Number of Solution Iterations	Number of simulated annealing iterations to run
Max Number of Annealing Iterations	Maximum number of simulated annealing iterations to perform in each solution iteration loop (optional)
Bridge Parameter Solution Mask	Mask of 0's and 1's indicating which target parameters to adjust in each simulated annealing iteration
STF Parameter Solution Mask	Mask of 0's and 1's indicating which OLI STF parameters to adjust in each simulated annealing iteration
MTF System Table File	File containing simulated annealing initial values
Band List	Bands included in fit parameter lists
Bridge Parameter Default Values (per band)	Initial values, overridden by target definition file
Water Reflectance	Water background signal (DN) level
Water Asymmetry	Asymmetry in water signal
Ground Sample Distance (in meters)	Band nominal GSD
Bridge Reflectance	Bridge signal (DN) level (all spans)
Span Width (meters)	Bridge span width (all spans)
Span Offset (meters)	Offset from the target centerline (all spans)

Algorithm Inputs	Algorithm/sub-algorithms
STF Parameter Default Values (per band)	Initial values for OLI STF model parameters
Gaussian Optical Blur (in microradians)	Optical blur (Gaussian) dimension
XT Detector Size (in microradians)	Cross-track detector dimension
AT Detector Size (in microradians)	Along-track detector dimension
Integration Time (in milliseconds)	Detector integration time
Exponential Decay (in microradians)	ALI heritage charge diffusion distance (not used)
Model/Data Phase Shift (in meters)	Model profile/data profile registration offset
Bridge Parameter Simplex Values (per band)	Parameter variation magnitudes for 6 bridge subfields
Same 6 subfields as default values	Same units as default values
STF Parameter Simplex Values (per band)	Parameter variation magnitudes for 6 STF subfields
Same 6 subfields as default values	Same units as default values

4.2.13.3.4 MTF Perform Inputs

Algorithm Inputs	Algorithm/sub-algorithms
ODL Parameter File	Processing parameters
MTF Estimate Trending File Name	File containing input trending records
Spatial Performance Report File Name	Output summary report file name
Band List	Bands to process
Number of Samples	Number of samples to use in synthesizing OLI edge spread function (ESF)
Oversampling Factor	Oversample factor (relative to nominal band GSD) to use in synthesizing OLI ESF (need not be the same as the estimation oversampling factor)
MTF Estimate Trending File	File containing trending output from one or more MTF Estimate runs
Contains one trending record for each target/band analyzed	See MTF Estimate Output table for trending record field definitions

4.2.13.4 Outputs

The MTF Bridge Characterization algorithm and its component sub-algorithms create the outputs listed in the following tables.

4.2.13.4.1 MTF Extract Outputs

Pixel Table Output Files	One ASCII output file containing the pixel records for each band, polygon, and target: "TarNPolyMBandKScaL.dat" Where: N is the target number M is the polygon number K is the band number L is the SCA number
Each pixel record contains:	Pixel record fields
Band number	Band number (enumerated type, 0 to 8)
SCA number	SCA number (1 to 14)
L1R line number	L1R line number where pixel was extracted
L1R sample number	L1R sample number where pixel was extracted
UTM Y coordinate (meters)	UTM Y coordinate of pixel

UTM X coordinate (meters)	UTM X coordinate of pixel
Ground velocity (meters/second)	Pixel ground velocity (used to scale integration time)
Pixel orientation/azimuth angle (radians)	Pixel azimuth angle (from UTM grid north)
Target range (meters)	Pixel target range (used to scale angular dimensions)
Pixel signal level (DN)	Pixel image value

4.2.13.4.2 MTF Profile Outputs

Target Profile Output Files	One ODL output file containing the oversampled profile for each band and target: "Profile_TarNBandK.odl" Where: N is the target number K is the band number
Each file contains:	ODL fields contained in each output Profile file.
Target Name	Name of bridge target from target definition file
Target ID	Target number (index) within scene
Band Number	Band number for current file
Number of Samples in Profile	Length of target profile
Ground Sample Distance (in meters)	Nominal ground sample distance for current band
Oversampling Factor	Oversample factor for constructed profile
Target Azimuth Angle (in radians)	Azimuth of target relative to pixels (target orientation)
Target Range (in meters)	Average range to target (to scale angular dimensions)
Ground Velocity (in meters/second)	Average target velocity (to scale integration time)
Profile Samples (mean pixel signal values)	Constructed target profile sample sequence
Weights (profile bin standard deviations)	Standard deviations of mean profile values

4.2.13.4.3 MTF Estimate Outputs

MTF Bridge Characterization Report File	Text report summarizing the MTF estimation fit results
Best Fit Modeled Profile Output Files	One fitted profile output file per target and band containing comma-delimited model profile values: "ModelProfile_TarNBandK.odl" Where: N is the target number K is the band number
MTF Bridge Characterization Trending File	Fit parameter solution summary containing one record for each target and each band
Each record contains:	Pipe (|) delimited fields in each trending record
Year	Year of acquisition
Day of Year	Day of year of acquisition
WRS Path	WRS path of acquisition
WRS Row	WRS row of acquisition
Target Name	Name of bridge target
Band Number	Band number for current record
Target Orientation Angle (in radians)	Target orientation relative to image pixels
Gaussian Optical Blur (in microradians)	Optical blur (Gaussian) dimension
XT Detector Size (in microradians)	Cross-track detector dimension

AT Detector Size (in microradians)	Along-track detector dimension
Integration Time (in milliseconds)	Detector integration time
Exponential Decay (in microradians)	ALI heritage charge diffusion distance (not used)
Model/Data Phase Shift (in meters)	Model profile/data profile registration offset
Final RMS Fit	RMS fit of model to data
Ground Sample Distance (in meters)	Current band nominal GSD
Target Range (in meters)	Range to target used to scale angular units
Ground Velocity (in meters/second)	Velocity used to scale integration time
Water Reflectance	Water background signal (DN) level
Water Asymmetry	Asymmetry in water signal
Number of Spans	Number of spans in target
For each span:	Fields for each target span
Bridge Reflectance	Bridge signal (DN) level
Span Width (meters)	Bridge span width
Span Offset (meters)	Offset from the target centerline

4.2.13.4.4 MTF Perform Outputs

OLI Spatial Performance Report File	Text report summarizing the MTF estimation fit results
For each band:	Following information is provided for each band
Summary of scenes/targets analyzed	List of each scene/target included
Final fitted 2D STF parameters	Final along- and across-track STF parameter values
Computed ES and HEE performance	Edge slope and half edge extent in each direction
Spatial performance summary table	Edge slope results vs. KPR thresholds (L8 only)
XT Synthesized ESF Profile File (per band)	One synthesized cross-track ESF output file per band containing comma-delimited normalized ESF values: “XT_ESF_BandK.odl” Where: K is the band number
AT Synthesized ESF Profile File (per band)	One synthesized along-track ESF output file per band containing comma-delimited normalized ESF values: “AT_ESF_BandK.odl” Where: K is the band number

4.2.13.5 Options

Heritage “Numerical Recipes in C” vs. GNU Scientific Library (GSL) simulated annealing implementation selection switch.

4.2.13.6 Procedure

MTF characterization generates an on-orbit estimate of the spatial (MTF and ESF) performance for the OLI instrument. The characterization estimates the OLI system transfer function (STF) and from this derives the corresponding edge spread function (ESF) from which the OLI edge slope and half edge extent key spatial performance parameters are determined. The STF is estimated by comparing a simulated modeled target to actual image data. The instrument model consists of optics, detector, sample phase, and ground sample distance components. The along-track model also contains an integration time component. The bridge target model uses several long bridges at different orientations with respect to the OLI ground track, each of which is described in a target definition file. The target list includes the two bridges within WRS path 22 row

39 that have historically been used for Landsat 7 spatial performance evaluation. The current implementation can support any bridge target that can be described in a target definition file.

The MTF characterization is accomplished by running four separate routines. The first three routines are applied to OLI scenes that contain bridge targets to derive results for those targets. The fourth routine summarizes the results of multiple targets to derive the full 2D OLI STF and corresponding spatial performance. A process flow diagram showing how these routines fit together is provided in Figure 4-49 below. The first routine, MTFEXTRACT, extracts radiometrically corrected but unresampled image pixels over the bridge targets. These pixels are selected based upon region of interest polygons assigned to each target. These polygons can be designed to avoid problematic portions of the target area. The extracted pixel data are written to a pixel table file as pixel records that also contain geolocation and viewing geometry information. The second routine, MTFPROFILE, creates an oversampled “super-resolution” profile of the bridge target(s) by calculating each pixel’s distance from the bridge target, and sorting the pixels into distance bins that are then averaged to form an oversampled representation of the target cross-section. The bins are designed to provide an oversampled profile, nominally with a 1/8 pixel granularity. Target profiles are written out as ODL files that include pixel standard deviations as well as mean values for each distance bin. This makes it possible to weight (or ignore) individual bins based upon the pixel content (or lack thereof) in each bin. The third routine, MTFESTIMATE, uses a simulated annealing method to minimize the difference between a theoretical target response synthesized from models of the target and OLI STF, and the actual oversampled profile data.

Any one target can only provide a one-dimensional estimate of the OLI STF in the direction perpendicular to the target. Thus, each target observed provides a cross-section of the full two-dimensional OLI spatial response. The idea underlying this algorithm is that by observing targets at multiple angles, we can infer the full 2D STF from the 1D cross-sections. This is the function of MTFPERFORM, the fourth MTF characterization routine. The more target observations we have, the better this procedure should work, hence the desire to support many different targets by using a parameterized target model. New targets can be added by simply creating an appropriate target definition file. This makes it possible to generate a richer set of observations for estimating OLI spatial performance.

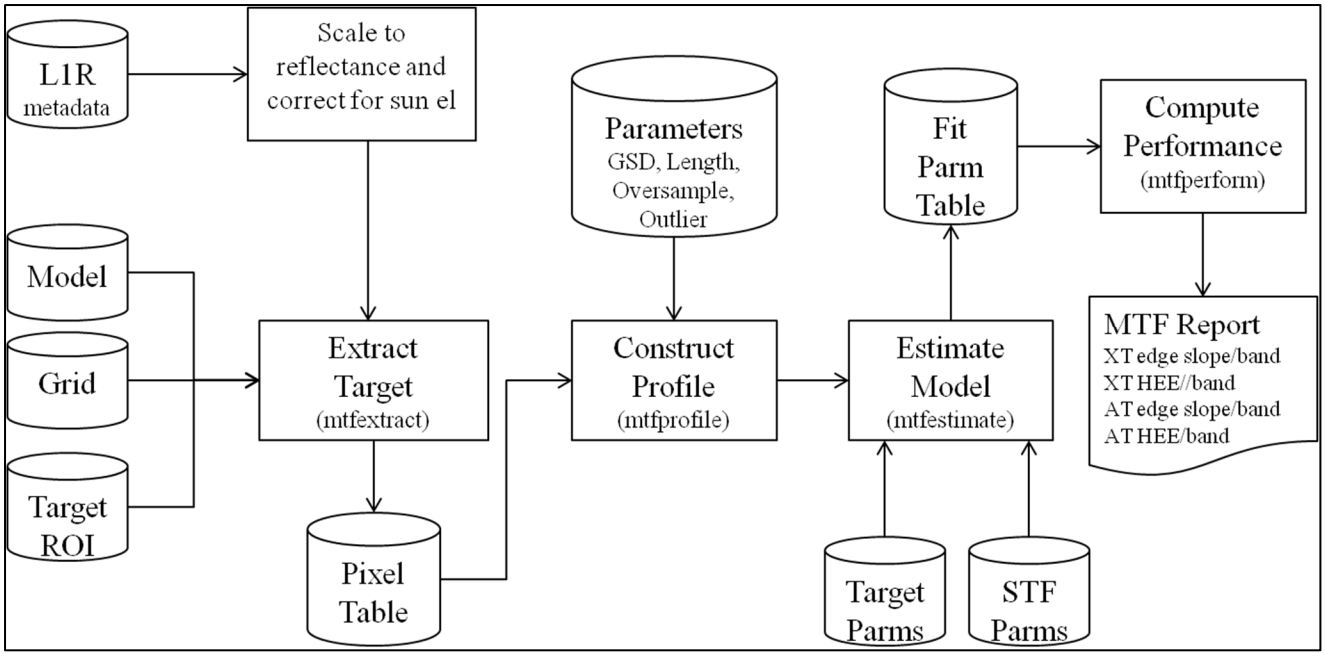

Figure 4-49. MTF Characterization Process Flow

The parameters required to define a bridge target include the following:

N_s = number of spans
For i = 1 to N_s:
w_i = span width (in meters)
x_i = span offset from target centerline (in meters)
ρ_i = span reflectance (per band) scaled to the L1R DN range
h_i = span height above water (in meters) (not currently used)
ρ_w = background (water) reflectance (per band) scaled to the L1R range
$\Delta\rho$ = background asymmetry (per band) in same units as ρ_w
H = water WGS84 ellipsoid height (in meters)
X_s,Y_s and X_e,Y_e – UTM coordinates of target centerline endpoints
N_a = number of polygon areas in the region of interest (ROI)
For j = 1 to N_a
N_p = number of points in area (first point is not repeated at end)
For k = 1 to N_p
$X_{j,k}$,$Y_{j,k}$ = UTM coordinates of polygon area j, vertex k

These parameters are packaged in an ODL target definition file. Simple example of a target definition file is shown below. Note that the file contains both the target model parameters and the target ROI.

```
OBJECT = MTF_Target
  BAND_LIST = (1,2,3,4,5,6,7,8,9)
  Number_of_Targets = 1
  Target0_Number_of_Spans = 1
  Target0_Name = "San_Mateo"
```

```
  Target0_Reflectance0 = (50.0, 100.0, 100.0, 100.0, 100.0, 35.0, 60.0, 50.0, 60.0)
  Target0_Water =          (20.0,  70.0,   70.0,   60.0,   45.0, 15.0, 15.0, 15.0, 15.0)
  Target0_Asymmetry =    ( 0.0,    0.0,     0.0,     0.0,     0.0,   0.0,   0.0,   0.0,   0.0)
  Target0_Span0 = 37.165
  Target0_Offset0 = 0.000
  Target0_Coordinates = (574409.169, 4163593.065, 567700.867, 4161024.250)
END_OBJECT = MTF_Target
OBJECT = MTF_ROI
  Target0_Number_Polygons = 4
  Target0_Polygon0_Vertices = 4
  Target0_Polygon0_Coordinates = (574194.604, 4164153.388, 574623.734, 4163032.742,
                                  573849.091, 4162736.108, 573419.961, 4163856.754)
  Target0_Polygon1_Vertices = 4
  Target0_Polygon1_Coordinates = (573380.402, 4163841.605, 573809.532, 4162720.959,
                                  572241.073, 4162120.348, 571811.943, 4163240.994)
  Target0_Polygon2_Vertices = 4
  Target0_Polygon2_Coordinates = (571775.697, 4163227.114, 572204.827, 4162106.468,
                                  569041.535, 4160895.146, 568612.405, 4162015.792)
  Target0_Polygon3_Vertices = 4
  Target0_Polygon3_Coordinates = (568568.810, 4161999.098, 568997.940, 4160878.452,
                                  567915.432, 4160463.927, 567486.302, 4161584.573)
END_OBJECT = MTF_ROI
END
```

Figure 4-50, Figure 4-51, and Figure 4-52 show the image sampling geometry relative to the bridge target. Figure 4-50 shows the image pixel sampling grid and the bridge target separately in a UTM north-up configuration. Figure 4-51 shows the image sampling pattern superimposed upon the target, and Figure 4-52 shows the sampled target rotated into the target orientation.

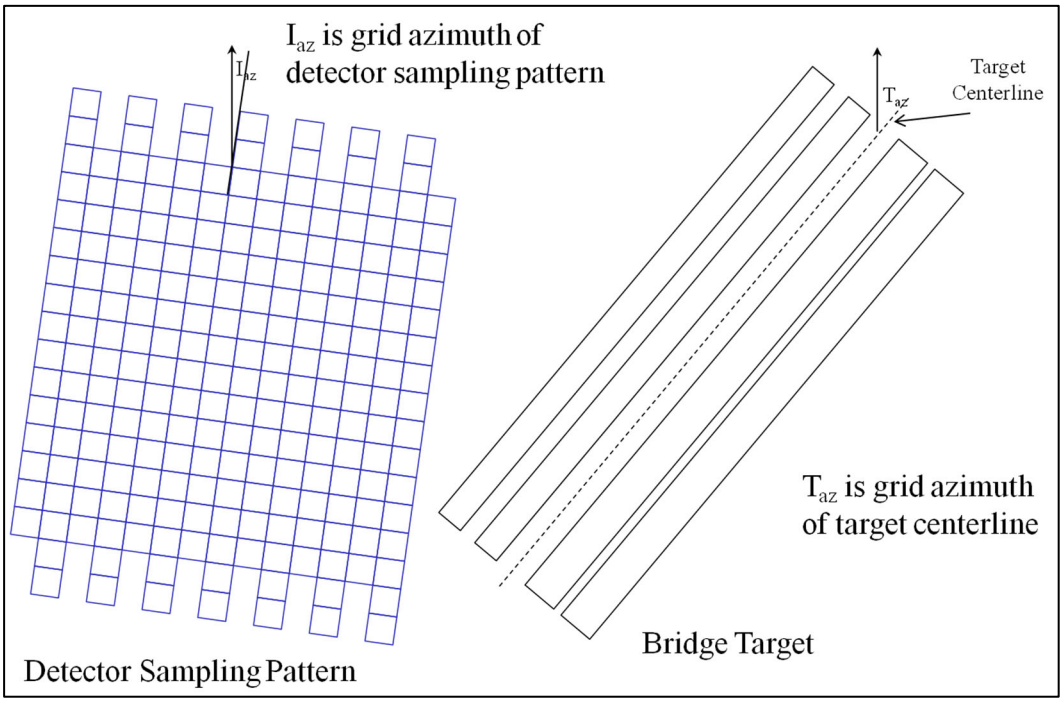

Figure 4-50. UTM North-Up Detector Pattern and Bridge Target

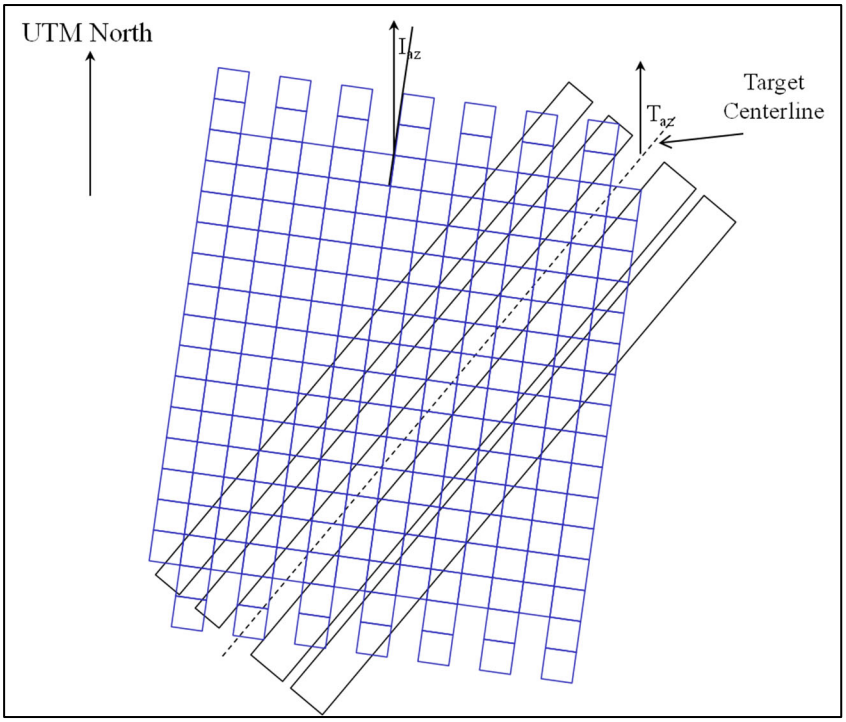

Figure 4-51. Superimposed Sampling Pattern and Bridge Target

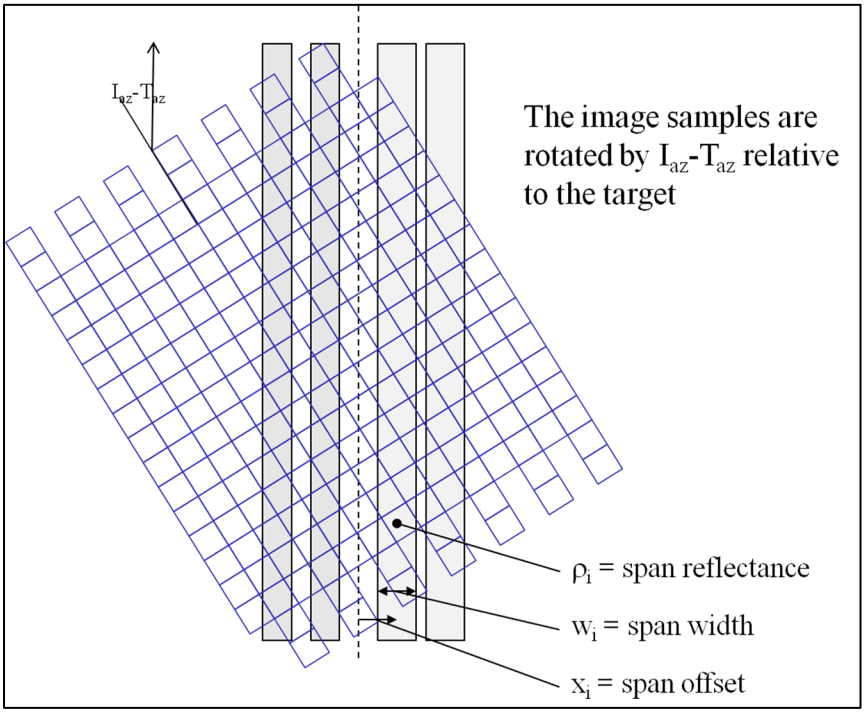

Figure 4-52. ***Pixel Sampling Pattern and Bridge Target in Target Orientation***

4.2.13.6.1 Phase 1. MTFEXTRACT: Extraction of image pixels covering the target area.

The first step in the MTF characterization process is to extract the unresampled (Level 1R) image pixels that sample the useful portion of the target area. Ideally the input image should be processed to top of atmosphere reflectance, but this is not strictly necessary. Note that not all portions of a target may be useful. For example, the Landsat 7 heritage targets, the Lake Pontchartrain Causeway and I-10 Bridge in WRS 22/39 are both double span bridges with periodic crossovers joining the spans for use as emergency vehicle turnaround areas. These crossovers are segments of the target bridge where the double-span target model is invalid. As such, the corresponding image pixels must be removed from the MTF analysis. This is controlled through the definition of the target region of interest, which can be designed to contain multiple polygons to exclude problematic portions of the target area.

Geometrically corrected (precision and terrain) version of the OLI image is generated as an aid locating the bridge target areas. Although the corrected L1TP image itself is not used, the precision model and grid used to generate it provide the information needed to project the target model into Level 1R image space. The ROI UTM coordinates are mapped back to the L1R image using the precision grid to locate the window for image pixel extraction. The procedure used to extract the desired image pixels is as follows:

For each band to be analyzed:

Since target definition files may contain more than one target for the same path/row, the procedure loops through the targets in the target definition file:

1. Each ROI polygon for the current target is examined to determine which SCAs contain usable pixels.
   1.1 Convert each vertex in the polygon from UTM X,Y to L1TP line, sample:

   l1g_line = (upper_left_y – target_y) / proj_distance_y
   l1g_sample = (target_x-upper_left_x) / proj_distance_x

   Where:
   upper_left_y = upper-left Y projection coordinate of imagery (from grid)
   upper_left_x = upper-left X projection coordinate of imagery (from grid)
   proj_distance_x = projection distance of imagery in X direction (from grid)
   proj_distance_y = projection distance of imagery in Y direction (from grid)
   target_x = UTM X location of ROI polygon vertex (from target definition file)
   target_y = UTM Y location of ROI polygon vertex (from target definition file)
   l1g_line = L1TP image line location corresponding to ROI vertex
   l1g_sample = L1TP image sample location corresponding to ROI vertex

   1.2 For each SCA, map the l1g_line and l1g_sample output space (L1TP) coordinates through the precision grid (using ols2ils) to obtain the corresponding input space (L1R) l1r_line and l1r_sample coordinates.

1.3 If any vertices fall inside the SCA, mark this SCA as active for the current target.

2. For each SCA that contains valid pixels:
   2.1 Open the L1R image for the current band, SCA.
   2.2 Open an output pixel table file for the current target, polygon, band, SCA.
   2.3 Compute the minimum bounding rectangle (MBR in L1R line/sample coordinates) of the current polygon in the current SCA.
   2.4 Loop through the MBR line/sample range:
   2.5 Perform a point-in-polygon test to determine if the current L1R line/sample falls inside the polygon.
   2.6 If it does not, go to the next point.
   2.7 Otherwise, use the model to compute the ground latitude/longitude of the current pixel using the target elevation as the height coordinate.
   2.8 Convert the latitude/longitude to UTM X/Y.
   2.9 Subtract 1 from the line number and repeat steps 2.7 and 2.8 above to compute the UTM X/Y coordinates of the same detector in the previous line (X_1,Y_1).
   2.10 Add 1 meter to the target elevation and repeat steps 2.7 and 2.8 above to compute the UTM X/Y coordinates of a point 1 meter higher along the same line of sight. Subtract the original X/Y coordinates to yield the change in X and Y per unit height: dX/dh and dY/dh
   2.11 Compute the grid azimuth of the pixel grid:
   $I_{az} = \text{atan2}(X_1 - X, Y_1 - Y)$
   2.12 Compute the ground velocity:
   $V_g = \text{sqrt}((X_1 - X)^2 + (Y_1 - Y)^2) / \text{line\_time}$
   2.13 Compute the distance from the sensor to the target using the spacecraft position **P** (OLI Line-of-Sight Projection / Grid Generation Algorithm) and the computed ground point position **X**:
   $D = | \mathbf{X} - \mathbf{P} |$
   2.14 Compute the line-of-sight zenith angle in the direction perpendicular to the target, using the target grid azimuth computed from the target centerline coordinates T_{az}:
   $\beta = \text{atan}(dX/dh * \cos(T_{az}) - dY/dh * \sin(T_{az}))$
   2.15 Compute the cosine of the OLI viewing angle, which is just the Z component of the OLI line-of-sight (OLI Line-of-Sight Projection / Grid Generation Algorithm):
   $\cos(\alpha) = los.z$
   2.16 Compute the effective target range for scaling angular units (e.g., IFOV) to ground meters (see Figure 4-53):
   $R = D \cos(\alpha) / \cos(\beta - \alpha)$
   2.17 Write the band, SCA, L1R line, L1R sample, UTM X, UTM Y, V_g, I_{az}, R, and the pixel DN value to the output pixel table file.

Note that a separate output pixel table file is created for each target, polygon, band, and SCA that contains usable data. Table 4-32 shows sample pixel table output.

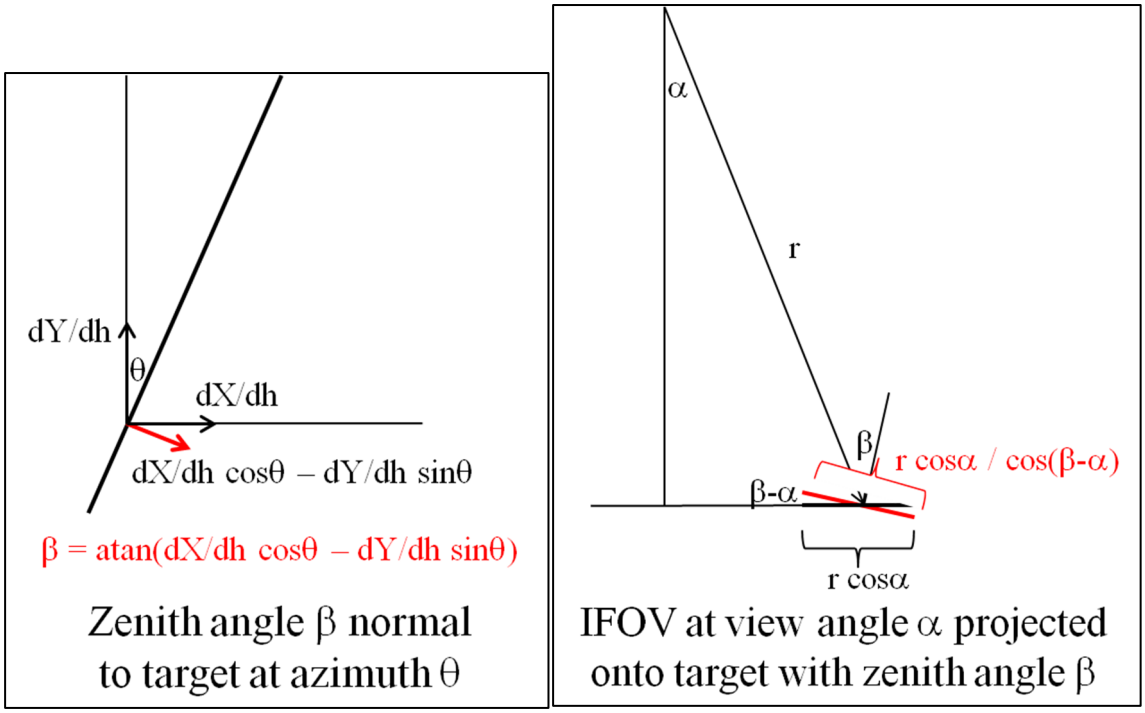

Figure 4-53. Geometry of Scaling IFOV in Angular Units to Ground Meters

Band	SCA	Line	Sample	UTM_Y	UTM_X	Velocity	Azimuth	Range	DN
3	1	4020.000	403.000	2900701.181185	424260.429036	6834.884661	0.220284	706065.972359	46
3	1	4020.000	404.000	2900753.848742	424301.915342	6834.886712	0.220284	706066.347523	48
3	1	4020.000	405.000	2900689.872665	424317.441281	6834.887905	0.220284	706065.348473	45
3	1	4021.000	403.000	2900672.948225	424254.048016	6833.106791	0.222279	706065.949158	46
3	1	4021.000	404.000	2900725.615771	424295.534335	6833.108760	0.222278	706066.324314	46
3	1	4021.000	405.000	2900661.639691	424311.060258	6833.110150	0.222279	706065.325258	46
3	1	4021.000	406.000	2900714.307151	424352.545791	6833.112119	0.222278	706065.701224	47
3	1	4021.000	407.000	2900650.331112	424368.070926	6833.113508	0.222279	706064.701945	46
3	1	4021.000	408.000	2900702.998484	424409.555673	6833.115477	0.222278	706065.078712	44
3	1	4021.000	409.000	2900639.051627	424425.086519	6833.116867	0.222279	706064.079550	44
3	1	4021.000	410.000	2900691.689772	424466.563981	6833.118836	0.222278	706064.456787	44
3	1	4021.000	411.000	2900627.742955	424482.094039	6833.120225	0.222279	706063.457398	44
3	1	4021.000	412.000	2900680.381013	424523.570715	6833.122195	0.222278	706063.835436	45
3	1	4021.000	413.000	2900616.434236	424539.099984	6833.123583	0.222279	706062.835829	45
3	1	4021.000	414.000	2900669.072208	424580.575875	6833.125553	0.222278	706063.214678	43
3	1	4021.000	415.000	2900605.119170	424596.132535	6833.126943	0.222279	706062.214534	43
3	1	4021.000	416.000	2900657.763357	424637.579462	6833.128911	0.222278	706062.594505	44

Table 4-32. Sample Pixel Table Output

4.2.13.6.2 Phase 2. MTFPROFILE: Create the oversampled target profile(s)

The second step in the MTF characterization process is to produce an oversampled "super-resolution" profile of each target for each image band. This is done by calculating the distance from each sample to the centerline of the target (preserving sign based on which side of the target the sample falls on), rescaling this distance from ground meters to profile samples by dividing by the nominal image GSD and multiplying by the desired

oversampling factor, collecting all of the samples that fall in each profile bin and computing a bin average DN value (after removing outliers), ordering the bin averages to form the output profile, and computing target average orientation angle, range, and ground velocity values for subsequent use by mtfestimate.

The profile construction procedure is as follows:

Load the target definition file for the current path/row.

For each target:
Calculate the grid azimuth of the target:

$T_{az} = \text{atan2}(X_e - X_s, Y_e - Y_s)$

Where: X_s, Y_s are the UTM coordinates of the target centerline start point
X_e, Y_e are the UTM coordinates of the target centerline end point

For each band:

1. Read all the pixel records over all SCAs and polygons, for this target and band.
2. For each pixel record, calculate the perpendicular distance to the target:
   a. Construct the vector from the centerline start point to the pixel:
      $v_i = [(X_i - X_s) \quad (Y_i - Y_s) \quad 0]^T$
   b. Construct the target centerline vector:
      $v'_t = [(X_e - X_s) \quad (Y_e - Y_s) \quad 0]^T$
   c. Normalize to form the corresponding target centerline unit vector:
      $v_t = v'_t / | v'_t |$
   d. Define the Z direction unit vector:
      $v_z = [0 \quad 0 \quad 1]^T$
   e. Calculate the distance by taking the cross product of the pixel vector with the centerline unit vector, and dotting the result with the Z direction unit vector:
      $D_i = v_z \bullet (v_i \times v_t)$
      The dot product extracts the Z component of the cross-product vector. Note that this calculation preserves the sign of the distance with points to the "right" of the target having positive distances and points to the "left" of the target having negative distances. Clearly, the sign convention depends upon the definition of the centerline start and end points.
   f. Convert the signed distance to a bin number:
      $B_i = \text{round}(D_i * O_s / GSD + N/2)$

   Where: O_s is the specified oversampling factor
   GSD is the nominal GSD for this band
   N is the specified length of the profile (number of bins)
3. For each pixel record, compute the target orientation relative to the pixel:
   $\theta_i = I_{az} - T_{az}$
4. For each bin:
   a. Calculate the mean and standard deviation of the pixel DN values in the bin and perform a t-distribution outlier rejection test based upon the user-specified outlier threshold.

i. If the DN farthest from the mean exceeds the t-distribution threshold, flag it as an outlier and recompute the statistics without that point.
ii. Continue removing the farthest point until all points pass the t-distribution outlier test.

b. Record the DN mean and standard deviation for all non-outlier points in the bin.

5. Compute the average, θ_{mean}, of the θ_i target orientation angle values for all non-outlier points across all bins.
6. Compute the average R_{mean} of the R_i (range) values for all non-outlier points across all bins.
7. Compute the average V_g of the V_i (ground velocity) values for all non-outlier points across all bins.
8. Write the output profile ODL file containing target name (from the target definition file), the target ID (number), the band number, the profile length (N), the oversampling factor (O_s), the nominal GSD, the target azimuth (θ_{mean}), the target range (R_{mean}), the ground velocity (V_g), the N-element profile data vector of mean DN values, and the N-element vector of bin standard deviations.

Note that this procedure creates one output profile ODL file for each target and each band. Figure 4-54 shows a sample profile ODL file.

```
OBJECT = TARGET_PROFILE
 TARGET_NAME = "King_Fahd_Causeway_W"
 TARGET_ID = 0
 BAND = 4
 NSAMP = 128
 GSD = 30.000
 OVERSAMP = 8
 TARGET_AZIMUTH =  1.515855155
 TARGET_RANGE = 706080.268
 GROUND_VELOCITY = 6832.106
 PROFILE = ( 555.888889, 579.500000, 573.000000, 574.000000, 575.416667, 564.200000, 576.833333, 569.300000, 571.100000, 567.222222, 565.333333,
       572.800000, 574.000000, 576.833333, 571.000000, 574.400000, 579.700000, 571.000000, 567.909091, 576.666667, 579.700000, 576.916667,
       574.400000, 567.600000, 584.800000, 569.300000, 569.666667, 574.272727, 571.333333, 581.400000, 565.900000, 567.700000, 564.200000,
       581.400000, 572.500000, 576.833333, 575.727273, 588.333333, 577.500000, 577.900000, 571.100000, 563.714286, 578.000000, 569.454545,
       574.000000, 559.000000, 554.000000, 553.875000, 560.800000, 560.181818, 566.625000, 618.500000, 615.750000, 647.300000, 701.000000,
       709.500000, 763.625000, 830.583333, 888.500000, 921.750000, 973.500000, 988.400000, 1041.800000, 1062.600000, 1015.333333, 1038.444444,
       985.083333, 928.833333, 852.500000, 934.500000, 801.300000, 761.600000, 695.250000, 640.333333, 653.888889, 617.333333, 584.800000,
       562.250000, 562.500000, 565.900000, 558.166667, 578.166667, 576.666667, 580.666667, 574.400000, 574.500000, 575.375000, 582.666667,
       585.181818, 585.500000, 585.333333, 582.333333, 583.285714, 572.700000, 581.200000, 578.100000, 582.666667, 575.727273, 585.500000,
       581.400000, 565.333333, 578.285714, 579.700000, 586.916667, 571.000000, 582.666667, 580.545455, 577.900000, 571.000000, 576.666667,
       583.428571, 575.250000, 577.181818, 572.800000, 588.333333, 572.888889, 579.600000, 572.800000, 578.000000, 578.428571, 575.333333,
       573.916667, 572.888889, 562.500000, 578.666667, 576.300000, 575.333333, 557.400000 )
 SIGMAS = ( 10.215729, 12.020815, 21.765799, 24.162330, 20.926097, 15.205262, 14.288690, 12.543701, 18.131924, 7.496295, 9.814955, 18.902087,
       19.249557, 17.049838, 12.020815, 14.223220, 21.929432, 11.333333, 16.688047, 9.814955, 18.672916, 23.635329, 10.751744, 14.223220, 14.584238,
       12.543701, 18.597817, 20.164776, 30.022214, 24.604426, 11.474125, 17.795443, 15.205262, 15.598077, 18.583962, 19.894418, 17.326805, 30.022214,
       12.961481, 14.548387, 16.264822, 9.086882, 19.735754, 9.169118, 17.832555, 24.289916, 24.041631, 25.892291, 18.274451, 15.715077, 33.027856,
       25.890796, 29.942748, 26.038646, 50.497525, 24.748737, 32.539811, 39.083729, 58.670885, 69.244391, 59.107247, 63.972564, 44.534132, 43.169434,
       43.775945, 48.179641, 59.689284, 58.203145, 75.500237, 48.790368, 49.647759, 55.869292, 48.392571, 46.112182, 40.489025, 46.245065, 30.367380,
       29.158434, 12.020815, 32.460061, 20.831939, 15.467463, 9.814955, 19.538424, 17.557525, 22.618822, 15.352408, 26.407070, 17.051793, 21.952428,
       17.803643, 9.814955, 13.136644, 12.543701, 8.778762, 21.926392, 26.407070, 15.569784, 19.332027, 18.566397, 9.814955, 13.375528, 16.865810,
       17.333406, 8.500000, 26.407070, 20.992206, 16.609569, 16.027754, 9.814955, 19.303343, 12.814232, 13.753677, 15.127605, 17.502381, 13.289511,
       14.645439, 17.119190, 19.735754, 13.709573, 16.576727, 16.115821, 17.919573, 12.020815, 15.215124, 18.366939, 16.576727, 14.223220 )
END_OBJECT = TARGET_PROFILE
END
```

Figure 4-54. Sample Output Profile ODL File

4.2.13.6.3 Phase 3. MTFESTIMATE: Fit target and STF models to profile data

The MTFESTIMATE routine consists of an initialization stage and a model parameter estimation loop. During the initialization phase, processing parameters are read from the input ODL parameter file, initial STF model parameters are read from a system table file, and the target model parameters are read from the target definition file. The processing parameters include arrays of flags that determine which variables to solve for in each iteration of the solution procedure. In addition to the STF model parameters, the system table file includes simplex bounds that control the behavior of the simulated annealing procedure that is used to solve for the STF and target model parameters. The output report and trending data files are also created during the initialization phase.

The model parameter estimation loop cycles through each target in the target definition file. For each target, the selected spectral bands are analyzed one-by-one. The MTFPROFILE routine will have created an oversampled target profile for each target/band combination. The best fit model parameters for each band and each target are found using a simulated annealing approach. Two user selectable implementations of the simulated annealing procedure are included, one provided by the GNU Scientific Library (GSL) and the other a modified version of the simplex downhill method described in “Numerical Recipes in C.” The latter is the heritage Landsat 7 implementation.

The simulated annealing algorithm allows for the solution of multi-variable non-linear functions. Multiple iterations of the annealing process are performed. Each iteration adjusts a user specified subset of the model parameters to best fit the oversampled target profile data. There are two models of functions involved in the process, one describing the bridge target and one describing the imaging system transfer function. Both the target and system (STF) models are formulated in the frequency domain. The target and system functions are evaluated in the frequency domain. The target function is then multiplied by the system transfer function in the frequency domain, producing a frequency domain representation of the simulated target profile. The inverse Fourier Transform is found of the simulated target response. This result is then compared against the oversampled target data profile generated in phase 2. The root mean square error between the synthesized data profile, produced from the models, and the measured data is used as the metric for the objective function minimization process in the simulated annealing procedure.

The Target Model

The bridge targets consist of one or more spans, each modeled as a rectangular pulse of a specified width at a specified offset from the target centerline, against a background (water) component. The background is allowed to be different on the two sides of the target by including an asymmetry term in the background model. The combined model is then the summation of the contributions from the individual bridge target spans and the background component. These target model components can be described in the frequency domain as follows:

$$Span_i : S(\omega) = (\rho_i - \rho_w) * \left(\frac{w_i * O_s}{GSD}\right) * sinc\left(\omega \frac{w_i}{2}\right) * e^{j\omega x_i}$$

$$Background : b(\omega) = N * \rho_w * \delta(\omega) + \Delta\rho * \frac{N}{2} * sinc\left(\omega \frac{D}{4}\right) * e^{-j\omega\frac{D}{4}}$$

$$\delta(\omega) = \begin{cases} 1\ if\ \omega = 0 \\ 0\ otherwise \end{cases}$$

Where:
ω = spatial frequency in radians per meter, evaluated at:
$\omega = O_s * 2\pi / GSD * i / N$ for i = -N/2 to (N/2 - 1)
w_i = bridge span i width (in meters)
x_i = span i offset from centerline (in meters)
O_s = oversampling factor
GSD = nominal image ground sample distance (in meters)
N = target sequence length in samples
D = target sequence length in meters = N * GSD / O_s
ρ_i = span i reflectance
ρ_w = background (water) reflectance
$\Delta\rho$ = background reflectance asymmetry (difference between bridge sides)
sinc(x) = sin(x)/x

$$j = \sqrt{-1}$$

The total target model is then the sum of the components above:

$$Target : T(\omega) = \sum_{i=}^{N_S} S_i(\omega) + b(\omega)$$

Where:
N_s = the number of spans in the target bridge model

The variable target model parameters are as follows:
the w_i span width values (one per span),
the x_i span offset values (one per span),
the ρ_i span reflectance values (one per span),
the ρ_w water reflectance value, and
the $\Delta\rho$ water reflectance asymmetry value.
The target physical dimensions, w_i and x_i, are held fixed while the reflectance values are allowed to vary and are typically input into the simulated annealing simplex solution as unknown variables to solve for.

The System Transfer Function Model

The OLI system transfer function model contains optical and detector components. The optical model includes both exponential and Gaussian components. The Gaussian component models optical blur and is the primary element of the optical sub-model. The exponential term was used in modeling the Advanced Land Imager (ALI) system transfer function and is included as a heritage element of the model, but is not used for OLI STF modeling. The detector model includes both detector dimension and integration time components. All of the model sub-components are included in both the along-track and across-track STF models except for the integration time component, which is only part of the along-track model.

The optical component models are formulated in the frequency domain as follows:

$$Optical\ (Gaussian)\colon O(\omega) = e^{\frac{-\omega^2 * \sigma^2}{2}}$$

Where:
σ=optical blur radius (in microradians) scaled to ground meters using R_{mean}, the target range parameter determined by MTFPROFILE

$$Exponential\colon C(\omega) = e^{-|\omega * d|}$$

Where:
d=exponential decay distance (in microradians) scaled to ground meters using R_{mean}, the target range parameter determined by MTFPROFILE

The detector dimension component model is formulated in the frequency domain using a sinc function:

$$Detector\colon D(\omega) = sinc\left(\frac{\omega * r}{2}\right)$$

Where
r = detector angular dimension (in microradians) scaled to ground meters using R_{mean}, the target range parameter determined by MTFPROFILE

The integration time model is also formulated using a sinc function. In this case the pulse dimension is the integration time scaled by the ground velocity, V_g.

$$Integration\colon I(\omega) = sinc\left(\frac{\omega * \tau * V_g}{2}\right)$$

Where
τ = detector integration time (in milliseconds) scaled to ground meters using V_g, the ground velocity parameter (in km/sec or m/msec) determined by MTFPROFILE

The combined across-track STF model is then the combination of the optical, exponential, and detector dimension components.

Cross-Track STF: $STF_x(\omega) = O(\sigma_x,\omega)\ C(d_x,\omega)\ D(r_x,\omega)$

Where:
σ_x = cross-track component of Gaussian blur
d_x = cross-track component of exponential decay
r_x = cross-track detector dimension

The along-track STF model is similar but includes the integration time component:

Along-Track STF: $STF_a(\omega) = O(\sigma_a,\omega)\ C(d_a,\omega)\ D(r_a,\omega)\ I(\tau,\omega)$

Where:
σ_a = along-track component of Gaussian blur
d_a = along-track component of exponential decay
r_a = along-track detector dimension
τ = detector integration time

Note that the integration time parameter has no directional index since it applies only in the along-track direction.

Synthesizing the Target Response
The modeled target response is synthesized from the component target and STF models by multiplying the system transfer function by the target model, computing the corresponding space domain representation by taking the inverse Fourier transform, and then comparing the resulting modeled target response to the measured target profile. There are two problems with this. The first is that the target model is a one-dimensional model that applies in the direction perpendicular to the bridge target whereas the STF model is a two-dimensional model that is separable into along- and across-track terms. We must determine the effective one-dimensional response of the 2D STF in the direction perpendicular to the bridge.

To do this, consider the relative rotation angle between the detector sampling pattern and the bridge target:

$\theta = I_{az} - T_{az}$

Where:
I_{az} is the detector sampling pattern UTM grid azimuth
T_{az} is the target UTM grid azimuth

The average θ angle for the target was computed by MTFPROFILE and is contained in the input oversampled profile ODL file. The STF model is formulated relative to the cross-track (ω_x) and along-track (ω_a) directions, so we express this coordinate system in terms of the target longitudinal (ω_{long}) and transverse (ω_{trans}) directions (see Figure 4-55):

$\omega_x = \omega_{trans} \cos\theta - \omega_{long} \sin\theta$
$\omega_a = \omega_{trans} \sin\theta + \omega_{long} \cos\theta$

In the longitudinal direction we are interested only in the DC frequency term since we seek to integrate away this direction. Therefore, setting $\omega_{long} = 0$ and $\omega_{trans} = \omega$ we can express ω_x and ω_a in terms of the transverse direction, which is the axis of the target model:

$\omega_x = \omega \cos\theta$
$\omega_a = \omega \sin\theta$

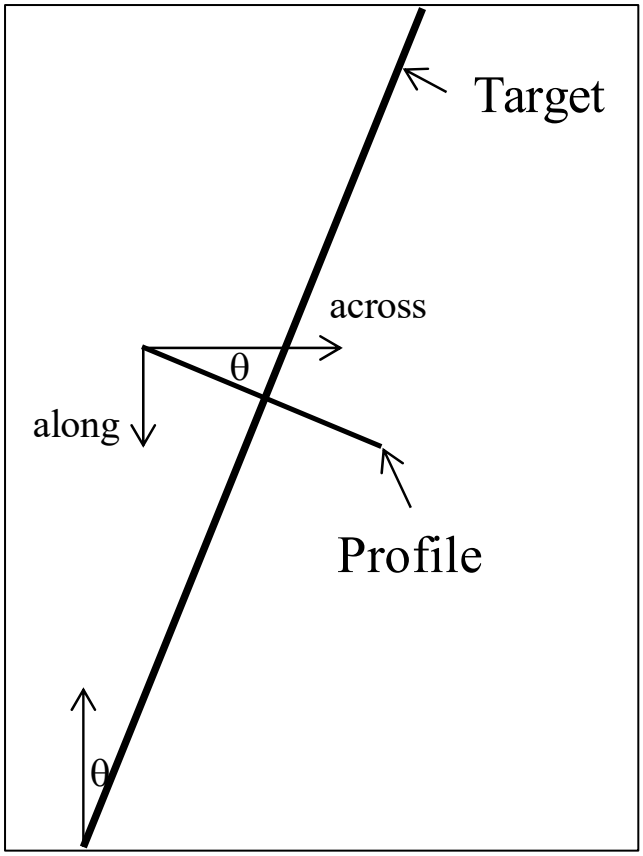

Figure 4-55. Bridge Cross-Section Profile at Angle θ Relative to Across-Track

These expressions can then be substituted into the STF model to yield the effective 1D STF in the target profile direction.

$STF_{trans}(\omega) = STF_x(\omega \cos\theta)\, STF_a(\omega \sin\theta)$

The second problem mentioned above is that we need an additional model component to register the synthesized analytical target response to the actual profile data, in the

transverse (across-bridge) direction. This registration component takes the form of a phase shift:

$$Phase: P(\omega) = e^{-j\ p}$$

Where:
p = scene dependent phase shift required to register the model to the profile data

The combined model used to synthesize the target response includes the transverse (1D) STF model, the target model, and the phase adjustment:

$M(\omega) = STF_x(\omega \cos\theta)\ STF_a(\omega \sin\theta)\ T(\omega)\ P(\omega)$

Substituting in the appropriate STF model components from above:

$M(\omega) = O(\sigma_x, \omega\cos\theta)\ D(r_x, \omega\cos\theta)\ C(d_x, \omega\cos\theta)$
$O(\sigma_a, \omega\sin\theta)\ D(r_a, \omega\sin\theta)\ C(d_a, \omega\sin\theta)\ I(\tau, \omega\sin\theta)\ T(\omega)\ P(\omega)$

For the time being we also assume that $\sigma_x = \sigma_a = \sigma$ so that:

$O(\sigma_x, \omega\cos\theta)\ O(\sigma_a, \omega\sin\theta) = O(\sigma, \omega)$

Substituting yields:

$M(\omega) = O(\sigma, \omega)\ D(r_x, \omega\cos\theta)\ C(d_x, \omega\cos\theta)$
$D(r_a, \omega\sin\theta)\ C(d_a, \omega\sin\theta)\ I(\tau, \omega\sin\theta)\ T(\omega)\ P(\omega)$

We will undo this simplifying assumption in phase 4: MTFPERFORM, but for this phase of the procedure we will transform $M(\omega)$ to the space domain as:

$M(x) = F^{-1}\{ M(\omega) \}$

And best fit M(x) to the measured profile data by adjusting the model parameters.

For the OLI system transfer function, the detector dimensions (r_x, r_a) and integration time (τ) are known to a high degree of precision, and the exponential decay model terms (d_x, d_a) are set to zero by definition. This only leaves the (single) Gaussian optics (σ) parameter, the phase (p) parameter, and the target model parameters as parameters subject to variation in the simulated annealing solution procedure.

The MTFESTIMATE model parameter solution procedure loops through each target specified in the target definition file. For each target we proceed as follows:

1. Preparing data for MTF estimate

1.1 Set up solution and model parameters. Initial values for both the bridge target, and the along-track and across-track STF parameters are read from the MTF system table file. The initial “step” values for the simplex routine are also loaded from the system table. The generic default target model parameters are then replaced with the bridge-specific values from the target definition file.

Loop on bands solving for the set of target and STF model parameters specified in the input ODL parameter file.

2. Read target profile produced from MTFPROFILE. This includes the target orientation angle, range scaling and ground velocity parameters in addition to the profile itself, as shown in Figure 4-55 above.

3. Perform parameter estimation.

3.1 Refine the bridge model parameter for background (water) by averaging DN values in profile generated in MTFPROFILE.

Loop on number of iterations of the simulated annealing process that is to be performed.

3.2 Set up bridge and system models needed for simulated annealing. Determine the model parameters to be solved for each iteration of the simulated annealing process. These are specified as input parameters. Set the step size to be used for each parameter in the simulated annealing process.

3.3 Determine the RMS fit for the initial model parameters.

The RMS is determined from the following steps:

3.3.1 Calculate the frequency domain response of the bridge target using the bridge model parameters (See The Target Model).

3.3.2 Calculate the frequency domain response of the system using the system model parameters (See The System Transfer Function Model).

3.3.3 Combine/multiply the target and the bridge responses, taking into account the phase shift parameter:

$$M(\omega) = \mathrm{STF}(\omega) * T(\omega) * e^{-\mathrm{j}*\omega*\mathrm{phase}}$$

Where
STF(ω) = system modeled transfer function

$T(\omega)$ = bridge target modeled transfer function
$M(\omega)$ = synthesized bridge profile produced from modeled bridge target and modeled system transfer function
phase = phase shift required to align model with data

3.3.4 Compute the inverse FFT of $M(\omega)$.

3.3.5 Calculate RMS between image data profile (produced from MTF profile) and modeled profile:

$$RMS = \sum_{n=0}^{NPTS} \left(M(n) - Image(n)\right)^2$$

Where
M = spatial domain transfer function of modeled target profile
Image = bridge profile from image data produced from MTFPROFILE
NPTS = number of points in MTF profile

3.4 Perform simulated annealing.

3.5 Check to see if optical and detector model parameters are positive. If any one of the parameters is not positive, take the absolute value of the parameter.

3.6 If necessary adjust the phase of the system transfer function to be within one half of a cycle of the bridge ground sample distance:

The solution of system transfer function returns a phase. This phase is periodic with respect to the number of over-sampled bins and the ground sample distance (N_{over}=N/2 for a profile length of N). Therefore the phase can be adjusted by multiples of N_{over}*GSD (ground sample distance).

```
while ( PHASE >= Nover*GSD )
        PHASE -=  2*Nover*GSD
while ( PHASE < -Nover*GSD )
        PHASE += 2*Nover*GSD
```

Where: PHASE = solution for phase component of system model

4. Write the results for this band to the report file.

4.1 Generate the final modeled spatial domain target response (profile) using the model parameters produced from the simulated annealing.

4.2 Write the modeled profile generated in 4.1 to an ASCII file.

5. Write the final fitted parameters to the output trending data file. Sample MTFESTIMATE trending output is shown in Figure 4-56.

Year	DOY	Path	Row	Target	Band	Azimuth	Optics	AT Det	XT Det	Integ	Phase	Fit	GSD	Range	Velocity	Water	Asym	NS	SRad	Width	Offset	SRad	Width	Offset
2000	261	022	039	Causeway	1	0.071766	2.3186e+01	42.303	40.863	3.600	-3.9371e-01	0.449	30.000	705744.375	6839.098	74.8	0.0	2	103.0	10.00	-17.20	98.3	10.00	17.20
2000	261	022	039	Causeway	2	0.071861	2.3335e+01	42.303	40.863	3.600	-4.6986e-01	0.439	30.000	705742.438	6839.057	74.8	0.0	2	103.3	10.00	-17.20	98.1	10.00	17.20
2000	261	022	039	Causeway	3	0.071831	2.3185e+01	42.303	40.863	3.600	2.4909e-01	0.590	30.000	705750.875	6839.282	54.1	0.0	2	86.3	10.00	-17.20	83.2	10.00	17.20
2000	261	022	039	Causeway	4	0.071698	2.3520e+01	42.303	40.863	3.600	-5.6739e-01	0.691	30.000	705748.562	6839.179	42.7	0.1	2	87.6	10.00	-17.20	82.7	10.00	17.20
2000	261	022	039	Causeway	5	0.071737	2.3719e+01	42.303	40.863	3.600	-1.3014e+00	0.342	30.000	705746.312	6839.127	12.8	0.0	2	36.4	10.00	-17.20	33.2	10.00	17.20
2000	261	022	039	Causeway	6	0.072200	2.2190e+01	42.303	40.863	3.600	-1.4280e+00	0.970	30.000	705756.188	6839.589	12.2	0.0	2	59.9	10.00	-17.20	55.5	10.00	17.20
2000	261	022	039	Causeway	7	0.072177	2.2230e+01	42.303	40.863	3.600	-1.2534e+00	0.742	30.000	705753.875	6839.504	11.4	-0.0	2	47.8	10.00	-17.20	46.5	10.00	17.20
2000	261	022	039	Causeway	8	0.071943	1.4302e+01	21.152	20.431	0.660	1.8103e+00	0.587	15.000	705740.812	6839.059	22.8	-0.1	2	72.3	10.00	-17.20	66.9	10.00	17.20
2000	261	022	039	Causeway	9	0.072099	2.2244e+01	42.303	40.863	3.600	-1.5391e+00	0.953	30.000	705758.938	6839.658	12.2	0.0	2	60.0	10.00	-17.20	55.5	10.00	17.20
2000	261	022	039	I-10_Old	1	-0.516236	1.7729e+01	42.303	40.863	3.600	-4.3485e+00	0.644	30.000	705829.562	6842.794	70.5	-0.0	2	89.0	14.20	-15.15	88.7	14.20	15.15
2000	261	022	039	I-10_Old	2	-0.516559	1.7895e+01	42.303	40.863	3.600	-3.9627e+00	0.677	30.000	705827.688	6842.774	70.6	-0.0	2	88.6	14.20	-15.15	89.1	14.20	15.15
2000	261	022	039	I-10_Old	3	-0.514778	1.7695e+01	42.303	40.863	3.600	-3.9253e+00	0.756	30.000	705836.250	6843.705	48.2	-0.0	2	70.8	14.20	-15.15	70.6	14.20	15.15
2000	261	022	039	I-10_Old	4	-0.515239	1.8426e+01	42.303	40.863	3.600	-4.1640e+00	0.982	30.000	705834.375	6843.275	36.8	0.0	2	68.7	14.20	-15.15	68.7	14.20	15.15
2000	261	022	039	I-10_Old	5	-0.515680	1.8586e+01	42.303	40.863	3.600	-4.1405e+00	0.452	30.000	705831.438	6842.912	12.2	-0.1	2	28.8	14.20	-15.15	28.9	14.20	15.15
2000	261	022	039	I-10_Old	6	-0.515707	1.6931e+01	42.303	40.863	3.600	-5.3443e+00	1.332	30.000	705841.750	6842.901	12.0	-0.1	2	50.4	14.20	-15.15	46.3	14.20	15.15
2000	261	022	039	I-10_Old	7	-0.515193	1.6656e+01	42.303	40.863	3.600	-4.4182e+00	1.046	30.000	705839.125	6843.217	11.3	-0.1	2	41.9	14.20	-15.15	39.5	14.20	15.15
2000	261	022	039	I-10_Old	8	-0.516498	1.2675e+01	21.152	20.431	0.660	-1.8332e+00	0.671	15.000	705825.938	6842.743	20.5	-0.2	2	54.7	14.20	-15.15	56.1	14.20	15.15
2000	261	022	039	I-10_Old	9	-0.516339	1.6710e+01	42.303	40.863	3.600	-4.6544e+00	1.160	30.000	705844.438	6842.754	12.0	-0.1	2	49.5	14.20	-15.15	48.0	14.20	15.15
2001	273	044	034	San_Mateo	1	-0.975840	1.6700e+01	42.303	40.863	3.600	-2.2189e+00	0.672	30.000	706792.125	6830.640	65.2	-0.0	1	80.6	37.17	0.00			
2001	273	044	034	San_Mateo	2	-0.977612	1.6851e+01	42.303	40.863	3.600	-2.2453e+00	0.824	30.000	706794.812	6829.192	65.3	-0.0	1	80.7	37.17	0.00			
2001	273	044	034	San_Mateo	3	-0.978794	1.7252e+01	42.303	40.863	3.600	-2.2222e+00	0.739	30.000	706790.438	6830.167	47.2	-0.0	1	64.0	37.17	0.00			
2001	273	044	034	San_Mateo	4	-0.976428	1.8472e+01	42.303	40.863	3.600	-1.3846e+00	1.084	30.000	706789.938	6831.141	32.2	0.1	1	56.9	37.17	0.00			
2001	273	044	034	San_Mateo	5	-0.975340	1.9445e+01	42.303	40.863	3.600	-1.3551e+00	0.479	30.000	706790.500	6831.211	9.3	0.0	1	21.7	37.17	0.00			
2001	273	044	034	San_Mateo	6	-0.983885	1.6970e+01	42.303	40.863	3.600	-3.7628e-01	0.965	30.000	706795.938	6827.138	9.1	0.0	1	30.4	37.17	0.00			
2001	273	044	034	San_Mateo	7	-0.982368	1.7032e+01	42.303	40.863	3.600	9.3589e-01	0.843	30.000	706792.938	6828.102	8.9	0.0	1	27.1	37.17	0.00			
2001	273	044	034	San_Mateo	8	-0.979730	1.0666e+01	21.152	20.431	0.660	-5.7014e-02	1.420	15.000	706798.375	6827.743	18.0	0.3	1	43.5	37.17	0.00			
2001	273	044	034	San_Mateo	9	-0.984056	1.6984e+01	42.303	40.863	3.600	-5.0594e-01	1.025	30.000	706800.000	6827.118	9.1	0.1	1	30.3	37.17	0.00			
2011	290	120	035	Qingdao	5	-0.807173	2.2016e+01	42.303	40.863	3.600	5.2081e+00	0.352	30.000	706503.562	6843.522	11.6	-0.1	1	15.3	34.50	0.00			
2011	290	120	035	Qingdao	6	-0.812448	2.0777e+01	42.303	40.863	3.600	5.9862e+00	0.575	30.000	706510.438	6838.578	11.5	-0.1	1	19.1	34.50	0.00			
2011	290	120	035	Qingdao	7	-0.812091	1.9721e+01	42.303	40.863	3.600	6.3677e+00	0.574	30.000	706507.500	6838.932	10.9	-0.1	1	17.1	34.50	0.00			
2011	290	120	035	Qingdao	8	-0.805650	1.0351e+01	21.152	20.431	0.660	8.6866e+00	0.536	15.000	706508.188	6843.890	20.8	0.1	1	26.1	34.50	0.00			
2011	290	120	035	Qingdao	9	-0.812184	2.0495e+01	42.303	40.863	3.600	5.5529e+00	0.601	30.000	706514.188	6838.801	11.5	-0.1	1	19.0	34.50	0.00			

Figure 4-56. Sample MTFESTIMATE Trending Output Data

4.2.13.6.4 Phase 4. MTFPERFORM: Calculate OLI Spatial Performance

The fourth and final phase of the MTF characterization process operates on the trended output produced by MTFESTIMATE, for a collection of bridge targets observed at different orientation angles. It exploits the different STF cross-sections observed by the various targets to separate out the OLI cross-track and along-track optical blur parameters.

Recall that a single Gaussian optical blur parameter value is estimated and recorded for each target analyzed by MTFESTIMATE, and that this blur value (σ_m) and the associated target azimuth (θ_m) are recorded in the trending data record for each band/target. These are the Azimuth and Optics columns shown in the sample output table of Figure 4-56. Each of these optical blur observations are taken to be a combination of the underlying along- and across-track optical blur parameters, with the combination determined by the target azimuth. Thus, each target record provides an observation of the form:

$$\cos^2\theta_m \, \sigma^2_x + \sin^2\theta_m \, \sigma^2_a = \sigma^2_m$$

Where:

σ_x is the across-track Gaussian optical blur parameter

σ_a is the along-track Gaussian optical blur parameter

Once multiple targets at different orientation angles have been measured we have a series of observations of this form, from which we can solve for σ^2_x and σ^2_a. Note that we must also apply constraints so that:

$\sigma^2_x >= 0$ and $\sigma^2_a >= 0$

making this a constrained least squares problem.

The unconstrained solution is:

$$\begin{bmatrix} \hat{\sigma}_x^2 \\ \hat{\sigma}_a^2 \end{bmatrix} = \begin{bmatrix} \sum_m \cos^2\theta_m \cos^2\theta_m & \sum_m \cos^2\theta_m \sin^2\theta_m \\ \sum_m \cos^2\theta_m \sin^2\theta_m & \sum_m \sin^2\theta_m \sin^2\theta_m \end{bmatrix}^{-1} \begin{bmatrix} \sum_m \cos^2\theta_m \sigma_m^2 \\ \sum_m \sin^2\theta_m \sigma_m^2 \end{bmatrix}$$

If the unconstrained solution produces $\sigma^2_x >= 0$ and $\sigma^2_a >= 0$, then the unconstrained solution is the solution. Otherwise:

If $\sigma^2_x < 0$ then:

$$\begin{bmatrix} \hat{\sigma}_x^2 \\ \hat{\sigma}_a^2 \end{bmatrix} = \begin{bmatrix} 0 \\ \sum_m \sin^2\theta_m \sigma_m^2 \Big/ \sum_m \sin^2\theta_m \sin^2\theta_m \end{bmatrix}$$

If $\sigma^2_a < 0$ then:

$$\begin{bmatrix} \hat{\sigma}_x^2 \\ \hat{\sigma}_a^2 \end{bmatrix} = \begin{bmatrix} \sum_m \cos^2\theta_m \sigma_m^2 \Big/ \sum_m \cos^2\theta_m \cos^2\theta_m \\ 0 \end{bmatrix}$$

The σ_m and θ_m values contained in the input trending records for each band to be analyzed are used to calculate the summations shown in the equations above. These equations are then used to solve for σ^2_x and σ^2_a, from which we then calculate σ_x and σ_a. The results for each band are summarized in the output spatial performance report file, shown below.

STF Fit Results for Band 1

Fit Based on 6 Trending Records:

Rec #	Year	DOY	WRS_Path	WRS_Row	Angle	Optics	Target
1	2000	261	022	039	0.071775	23.213	Causeway
2	2000	261	022	039	-0.515816	17.320	I-10_Old
3	2001	273	044	034	-0.975840	16.700	San_Mateo
4	2001	227	163	042	1.509833	20.330	King_Fahd_Causeway_W
5	2001	227	163	042	1.293609	19.314	King_Fahd_Causeway_C

6 2001 227 163 042 1.742302 21.356 King_Fahd_Causeway_E

OLI STF Fitted Parameters:
Cross-Track Optics: 20.524 urad Along-Track Optics: 19.430 urad
Cross-Track Detector: 40.863 urad Along-Track Detector: 42.303 urad
Charge Diffusion: 0.000 urad Integration Time: 3.600 msec

Having solved for the across-track and along-track optical blur parameters, we use these values of σ_x and σ_a to construct the full across- and along-track transfer functions STF_x and STF_a:

Cross-Track: $STF_x(\omega) = O(\sigma_x, \omega)\, D(r_x, \omega)\, C(d_x, \omega)$
$STF_x(\omega) = \exp(-\omega^2 \sigma_x^2/2)\, \mathrm{sinc}(\omega\, r_x/2)\, \exp(-|\omega\, d_x|)$

Along-Track: $STF_a(\omega) = O(\sigma_a, \omega)\, D(r_a, \omega)\, I(\tau, \omega)\, C(d_a, \omega)$
$STF_a(\omega) = \exp(-\omega^2 \sigma_a^2/2)\, \mathrm{sinc}(\omega\, r_a/2)\, \mathrm{sinc}(\omega\, \tau\, V_g/2)\, \exp(-|\omega\, d_a|)$

These models are then used to evaluate the corresponding OLI spatial performance characteristics.

Parameter Error Estimates

The unconstrained least squares solution can also be used to estimate the errors in the σ^2_x and σ^2_a, parameters determined from the solution. First, the residuals (v_i) must be computed for each of the input trending records used in the solution:

$$v_i = \sigma^2_i - (\cos^2\theta_i\, \sigma^2_x + \sin^2\theta_i\, \sigma^2_a)$$

using the best-fit values of σ^2_x and σ^2_a to solve for the difference between the measured blur values (σ^2_i) and the corresponding modeled values ($\cos^2\theta_i\, \sigma^2_x + \sin^2\theta_i\, \sigma^2_a$). The residuals are then used to compute the variance of an observation of unit weight:

$$\sigma^2_{UW} = \sum_{i=1}^{N} v_i^2 \Big/ (N-2)$$

where N is the number of trending record observations used in the solution and (N-2) is the number of degrees of freedom in the solution (number of observations minus number of parameters). The estimated parameter covariance matrix is then:

$$Cov = \sigma^2_{UW} \begin{bmatrix} \sum_m \cos^2\theta_m \cos^2\theta_m & \sum_m \cos^2\theta_m \sin^2\theta_m \\ \sum_m \cos^2\theta_m \sin^2\theta_m & \sum_m \sin^2\theta_m \sin^2\theta_m \end{bmatrix}^{-1}$$

The diagonal terms of this covariance matrix are the aposteriori parameter estimate variances. Note that since it is the optical blur variances (σ^2_x and σ^2_a) that were estimated, the covariance terms apply to the σ^2 values, not the σ values.

To add conservatism to the performance estimates used to evaluate the L8 OLI key performance requirement (KPR) relating to spatial response (KPR #4), we use the covariance matrix to compute the 90% confidence interval bounding the best-fit optical blur parameter estimates. We then use the 90% confidence interval limiting value that represents the best OLI spatial performance (i.e., smallest blur) that is consistent with the observed data to within the specified confidence. We do this by applying the 90% circular error scaling factor (2.146) to the estimate standard deviations, and subtracting the resulting value from the best fit parameter value:

$$\sigma_x^2(90\%) = MAX(\sigma_x^2 - 2.146 * \sqrt{Cov(1,1)}, 0)$$
$$\sigma_a^2(90\%) = MAX(\sigma_a^2 - 2.146 * \sqrt{Cov(2,2)}, 0)$$

The resulting 90% confidence values of $\sigma_x(90\%)$ and $\sigma_a(90\%)$ can then be used in the OLI STF model to compute conservative (best case) estimates for the OLI edge slope and edge extent parameters. Note that the fewer the available input records, the larger will be the difference between the best-fit results and the best case 90% confidence results. These additional calculations will not be necessary for the L9 OLI-2, since there are no formal key performance requirements for OLI-2, and only the best-fit results need be reported, although there is no harm in continuing to report both.

Calculating OLI Spatial Performance Parameters

The along- and across-track OLI system transfer function models are evaluated using the profile sequence length (N) and oversampling parameter (O_s) values specified in the MTFPERFORM input ODL parameter file. Note that these values need not be the same as the values used during the model estimation procedure. A higher resolution (larger N and O_s values) model would typically be used at this stage to increase the fidelity of the edge response model used to determine edge slope and half edge extent.

The along- and across-track STF models, which are formulated in sensor units, are scaled to ground units using the nominal Landsat altitude (705 km). They are then inverse Fourier transformed to yield the corresponding along- and across-track line spread functions (LSFs). These are then integrated to compute the along- and across-track edge spread/response functions (ESFs):

```
ESF(0) = LSF(0)
for i = 1 to N-1
        ESF(i) = ESF(i-1) + LSF(i)
```

Where:

ESF(i) = edge spread function at index i
LSF(i) = line spread function at index i

We next interpolate the X locations corresponding to the 5%, 40%, 50%, 60%, and 95% edge response points. Since, by construction, the ESF is normalized so that the minimum value is 0 and the maximum value is 1, this amounts to finding the ESF indices that straddle 0.05, 0.4, 0.5, 0.6, and 0.95 and then linearly interpolating the exact value:

```
for v in {0.05, 0.4, 0.5, 0.6, 0.95}
        i = 1
        while i < N and ESF(i) < v
                i = i+1
        pos(v) = (i-1) + (v – ESF(i-1)) / (ESF(i) – ESF(i-1))
```

We then calculate the edge slope from the pos(0.4) and pos(0.6) values:

Edge Slope = O_s * (0.6 – 0.4) / (pos(0.6) – pos(0.4)) / GSD

We calculate the half edge extent from the pos(0.05), pos(0.5), and pos(0.95) values:

Low half edge extent = GSD / O_s * (pos(0.5) – pos(0.05))
High half edge extent = GSD / O_s * (pos(0.95) – pos(0.5))
Half Edge Extent = MAX(Low half edge extent, High half edge extent)

The edge slope and half edge extent values determined for each band are summarized at the end of the output spatial performance report file, as shown below. Note that spatial performance estimates are computed using both the best-fit OLI STF parameter values and the best case 90% confidence interval values described above.

Edge Slope and Half Edge Extent Spatial Performance by Band

Band	NObs	XT_ES	XT_HEE	AT_ES	AT_HEE	ES_Spec	HEE_Spec
1	6	0.02338	27.451	0.02207	29.031	> 0.027	< 23.0
2	6	0.02317	27.709	0.02207	29.026	> 0.027	< 23.0
3	6	0.02301	27.924	0.02204	29.065	> 0.027	< 23.0
4	6	0.02273	28.287	0.02139	30.005	> 0.027	< 23.5
5	6	0.02245	28.652	0.02135	30.054	> 0.027	< 24.0
6	6	0.02379	26.953	0.02253	28.389	> 0.027	< 28.0
7	6	0.02399	26.708	0.02254	28.378	> 0.027	< 29.0
8	6	0.03813	16.955	0.03568	18.128	> 0.054	< 14.0
9	0	0.02399	26.708	0.02254	28.378	> 0.027	< 27.0

Best Case Edge Slope and Half Edge Extent Spatial Performance by Band

Band	NObs	XT_ES	XT_HEE	AT_ES	AT_HEE	ES_Spec	HEE_Spec
1	6	0.02540	25.112	0.02287	27.946	> 0.027	< 23.0
2	6	0.02516	25.370	0.02287	27.938	> 0.027	< 23.0
3	6	0.02480	25.775	0.02279	28.055	> 0.027	< 23.0
4	6	0.02432	26.318	0.02202	29.093	> 0.027	< 23.5
5	6	0.02384	26.890	0.02193	29.221	> 0.027	< 24.0
6	6	0.02562	24.878	0.02326	27.430	> 0.027	< 28.0
7	6	0.02580	24.687	0.02324	27.449	> 0.027	< 29.0
8	6	0.04271	15.089	0.03739	17.282	> 0.054	< 14.0
9	0	0.02580	24.687	0.02326	27.430	> 0.027	< 27.0

In the likely event that there are no trended output results for the cirrus band (band 9), the results for the other two short-wave infrared bands (band 6: SWIR1, and band 7: SWIR2) will be used to estimate cirrus performance. This is done solely for purposes of key performance requirement evaluation per the Key Performance Requirements Evaluation Plan (see below). To effect this, the cirrus band estimated edge slopes are taken to be the larger of the band 6 and band 7 results, whereas the cirrus band edge extents are taken to be the smaller of the band 6 and band 7 results:

Cirrus XT edge slope = MAX(SWIR1 XT edge slope, SWIR2 XT edge slope)
Cirrus AT edge slope = MAX(SWIR1 AT edge slope, SWIR2 AT edge slope)
Cirrus XT edge extent = MIN(SWIR1 XT edge extent, SWIR2 XT edge extent)
Cirrus AT edge extent = MIN(SWIR1 AT edge extent, SWIR2 AT edge extent)

4.2.13.7 Notes

Some additional background assumptions and notes include the following:

1. Bridge region of interest polygons are contained in the target definition file for each target WRS path/row. These polygons are designed to avoid undesirable areas such as bridge crossovers.
2. L1R file name is currently used to extract acquisition date.
3. Target model initialization parameters include; reflectance of bridge span(s) (assumed to be the same for all), reflectance of water, reflectance asymmetry, width of bridge span(s), bridge span offset(s), and GSD of bridge pixels.
4. System transfer function initialization parameters include; optical Gaussian component, XT detector size, AT detector size, integration time, exponential decay (=0), and phase.
5. Target simplex bounds include; reflectance of bridge span(s) (assumed to be the same for all), reflectance of water, reflectance asymmetry, width of bridge span(s), bridge span offset(s), and GSD of bridge pixels.
6. System transfer function simplex bounds include; optical Gaussian component, XT detector size, AT detector size, integration time, exponential decay (=0), and phase.
7. Implementation has the bridge and system models plus simplex bounds stored within a MTF systems file.
8. Contents of MTF bridge characterization report:
    a. Scene acquisition date and WRS path/row
    b. For each target and for each band;
        i. Geometric parameters: GSD, target range, velocity, and orientation
        ii. Target parameters: target name, number of spans, span radiances, water radiance, water asymmetry, span widths, span offsets
        iii. System transfer function parameters: optical Gaussian, AT detector size, XT detector size, integration time, exponential decay, phase.
        iv. Fit statistics: Root mean square error to model.

4.2.14 OLI Detector Map Verification (DMV) Algorithm

4.2.14.1 Background/Introduction

There is a need to verify that the detector map used to determine the active detectors on the Operational Land Imager (OLI) are set to their proper values. Early in the L8 mission, during the On-Orbit Verification (OIV) period, the detector map became corrupted by a failed load operation and a misalignment between detectors of the OLI products occurred. Since that time, operational changes have been implemented to resolve the issue causing that particular case and it is believed that these changes will be effective in preventing the detector map from becoming corrupted by the failed load mechanism in the future. However, these changes lead to a less frequent load of the detector map, thereby exposing the detector map to bit flips due to particle hits on the OLI electronics. This algorithm will detect an unexpected change to the L8 OLI or L9 OLI-2 detector map whether due to an isolated bit flip or to a failed detector map load. The algorithm has been developed and tested to determine scene by scene whether the even/odd detector alignment within a given Level-1R file matches that associated with the Calibration Parameter File (CPF) hence determining if a detector map selection has been set correctly or incorrectly. This prototype algorithm is implemented using a pair of applications: detmap and getmap. The first measures the offsets between detectors of a given Level-1R (detmap), the second determines if these measured offsets match the desired offset needed to align the detectors (getmap) for generating Level-1 datasets. Operationally, a third process would need to be implemented which would be a database query that assesses a group of verified datasets over a given time range. This step is also discussed within this algorithm description.

4.2.14.2 Dependencies

The geometric model, CPF, and Level-1R file must be present for the algorithm to run. Because of this ancillary data preprocessing, generation of L1R, and line-of-sight model creation must also have been run.

4.2.14.3 Inputs

The OLI DMV algorithm uses the inputs listed in the following table. Inputs for the detmap and getmap tools are listed separately. It is worth noting that values that are stored within the CPF that are needed for this algorithm are also stored in the geometric model. At times within the document CPF items are noted but they are retrieved from those CPF items stored within the geometric model.

Algorithm Inputs (detmap)
Band to Process
Geometric Model File
Output file name (per band)
Level-1R
Algorithm Inputs (getmap)
Band to process
Output from detmap
Output file name (per band)
Output file overall results

4.2.14.4 Outputs

It's possible that detmap and getmap could be combined into a one-step process however the outputs should remain the same: there should be two output files created, and both should produce a file for each OLI band. The current processing flow also pipes the standard output from getmap to a file that serves as the overall assessment report for the processing of that scene, per-SCA for each band, indicating whether and where an incorrect detector offset was found. With that in mind listed below are some key elements to this algorithm that should remain the same after implementation:

a) The step that measures the between-detector offsets and creates an output table (in a formatted ASCII file) of results for each band, along with the expected results, should be present. This is created and present in the current detmap step. A switch could be provided as to whether this output is created but the option still needs to be present.
b) The step that prints out (in a formatted ASCII file) the individual incorrect detectors, their current selection, and what its selection should have been for each band and SCA should remain. This is created and present in the current getmap step. A switch could be provided as to whether this output is created but the option still needs to be present.

There should be a file containing the overall summary of the results for all the bands for a given scene. This output is currently captured from the standard output from the current getmap step. This type of information should also be trended within the database. There should be a switch to turn this trending off.

4.2.14.5 Options

For the implementation of this algorithm there should be an option to enable or disable the report files.

4.2.14.6 Procedure

1) Get user input values; Level-1R imagery, geometric model, CPF, and output file name.
2) Read in Level-1R, CPF detector offsets, CPF nominal fill values, and Level-0 detector alignment fill offset values. The offsets and fill values are read from the geometric model. It is worth noting that currently all OLI Level-0 alignment offset fill values are zero.
3) For each band of imagery, read in one SCAs worth of imagery and determine offsets.
   3.1) Determine desired offsets to be measured based on OLI band number.

```
if( Level-1R band >= 6 )
      max offset =  10;
      min offset = -10;
 else
      max offset =  6
      min offset = -6
```

3.2) Loop through each detector and measure all possible offsets between this detector and its neighbor. These detectors are defined as the right and left detectors within the code (and within this document).

for det = 0 to det < number of detectors within band, det++
left detector = ldet = Level 1R [det]
right detector = rdet = Level 1R [det+1]
if(det %2 == 0)
start offset = -max offset
end offset = 2
else
start offset = 2
end offset = max offset

Where the Level-1R [] is indicating a detectors worth of data, the data from the start line to the end line, within a Level-1R file (i.e., a full column of image data).

Left fill = filll = Level-0 applied fill to left detector
Right fill = fillr = Level-0 applied fill to right detector
Left delay = delayl = CPF detector delay/offset for left detector
Right delay = delayr = CPF detector delay/offset for right detector
Nominal detector fill = nom_fill = CPF nominal detector fill
ndet = number detectors in SCA/band

3.2.1) For all possible offsets determine correlation coefficient.

For off = start offset to off <= end offset
Initialize statistic variables for correlation coefficient.
Sum left detector = suml = 0.0
Sum right detector = sumr = 0.0
Sum left detector squared = sumlsq = 0.0
Sum right detector squared = sumrsq =0.0
Sum of product of left and right detector = sumlr = 0.0
Number of sampled counted = count = 0

3.2.2) Sum statistic along two detectors.

for line = start offset to line < number lines within band – max offset, line++
suml = suml + Level-1R[det][line]
sumr = sumr + Level-1R[det+1][line+off]
sumlsqr = sumlsqr + Level-1R[det][line] * Level-1R[det][line]
sumrsqr = sumrsqr + Level-1R[det+1][line+off] * Level-1R[det+1][line+off]
sumlr = sumlr + Level-1R[det][line] * Level-1R[det+1][line+off]
count = count + 1

3.2.3) Calculate correlation coefficient for given detector and offset

Calculate correlation coefficient between left and right detector for the current right detector offset.

numer = ((double)count) * sumlr - suml * sumr

templ = ((double)count) * sumlsq - suml * suml
tempr = ((double)count) * sumrsq - sumr * sumr
if templ < 0.0 || tempr < 0.0
error calculating correlation coefficient
report an error and continue with detector.
templ = sqrt(templ)
tempr = sqrt(tempr)
corr = correlation coefficient = numer / (templ * tempr)

3.2.4) Determine if current location is the “best” measured offset:
If off = start offset
max[det] = maximum peak = corr
meas[det] = measured peak (location) = off
else (corr >= maximum peak)
max[det] maximum peak = corr
meas[det] measured peak (location) = off

4) Output the current SCA, detector number(left), peak location, maximum peak, nominal detector fill, Level-0 fill left pixel, Level-0R detector fill right pixel, left pixel detector offset, right pixel detector offset.
SCA
Left detector number (det)
Measured peak location (meas[det])
Maximum peak (maximum peak[det])
Nominal detector fill (nom_fill)
Level-0R left pixel fill – fill[det]
Level-0R right pixel fill – fillr[det]
CPF left pixel offset = delayl[det]
CPF right pixel offset = delayr[det]

5) Finished with detmap

Detector Map Verification Sub-Algorithm (getmap)

1) Loop through all SCAs.

1.1) Read/Get detmap information for one SCA of one band.
There is not valid detmap information for the last measurement. Fill that information with the 2nd to last measurement.
cpfpos[ndet-1] = (int) round(delayl[ndet-2])
cursca[ndet-1] = cursca[ndet-2]
curdet[ndet-1] = curdet[ndet-2] + 1
meas[ndet-1] = 0
Initialize band type (swir or not) for deselect offsets.
if (band == 6 || band == 7 || band == 9)
swir = 1

```
else
  swir = 0
```

1.2) Determine location of all primary sets of detectors based on measurements.

```
primary[0] = -99
for det=1, det<ndet-2, det++
          if   meas[det] == -2 && meas[det-1] == 2 && meas[det+1] == 2
               primary[det] = 0
          else if  meas[det] == 2 && meas[det-1] == -2 && meas[det+1] == -2
               primary[det] = 2
        else
             primary[det] = -99
 primary[ndet-2] = -99
 primary[ndet-1] = -99
```

1.3) Work forward determining offset positions not already assigned by 1.2)

```
forward[0] = primary[0]
   for det=1, det<ndet, det++
         if primary[det] > -90
              forward[det] = primary[det]
         else if ( forward[det-1] > -90 )
             forward[det] = forward[det-1] - meas[det-1]
        else
             forward[det] = -99
```

1.4) Work backwards from last detector determining offset positions not already assigned by 1.2) and 1.3).

```
 reverse[ndet-1] = forward[ndet-1];
 for  det=ndet-2, det >= 0, det--
       if   primary[det] > -90
            reverse[det] = primary[det]
     else if ( reverse[det+1] > -90 )
          reverse[det] = reverse[det+1] + meas[det]
     else
         reverse[det] = forward[det]
```

1.5) Scan through detectors assigning CPF and measured selections. The purpose of the following block of if statements is to provide a look up table (Table 4-33) designating the detector selection based on the CPF value, odd or even detector, and the band that is being processed. The flags follow the settings:

CPF selection = 0 for primary
CPF selection = 1 for secondary
CPF selection = 2 for tertiary

The following look up table is then established:

Even/Odd	SWIR Band	Offset	Selected Detector
even	Don't care	0	0
even	Don't care	-2	1
even	1	-4	2
odd	Don't care	2	0
odd	Don't care	4	1
odd	1	6	2

Table 4-33. Detector Select Look-Up

```
status = 0
for det=0, det<ndet, det++
      Initialize CPF detector selection.
          cpf_sel[det] = -1
      if ( curdet[det]%2 == 0 && cpfpos[det] == 0 )
          cpf_sel[det] = 0
     if ( curdet[det]%2 == 0 && cpfpos[det] == -2 )
         cpf_sel[det] = 1;
        if ( swir == 1 && curdet[det]%2 == 0 && cpfpos[det] == -4 )
            cpf_sel[det] = 2;
    if ( curdet[det]%2 == 1 && cpfpos[det] == 2 )
        cpf_sel[det] = 0;
    if ( curdet[det]%2 == 1 && cpfpos[det] == 4 )
       cpf_sel[det] = 1;
    if ( swir == 1 && curdet[det]%2 == 1 && cpfpos[det] == 6 )
       cpf_sel[det] = 2;
    if ( cpf_sel[det] < 0 )
          Error assigning CPF selection at: SCA and detector number

     Determine image detector selection.
    img_sel[det] = -1
    if curdet[det]%2 == 0 && reverse[det] == 0
        img_sel[det] = 0
    if curdet[det]%2 == 0 && reverse[det] == -2
       img_sel[det] = 1
    if swir == 1 && curdet[det]%2 == 0 && reverse[det] == -4
       img_sel[det] = 2
     if curdet[det]%2 == 1 && reverse[det] == 2
       img_sel[det] = 0
    if curdet[det]%2 == 1 && reverse[det] == 4
      img_sel[det] = 1
    if swir == 1 && curdet[det]%2 == 1 && reverse[det] == 6
      img_sel[det] = 2
     if reverse[det] < -90 || ( forward[det] > -90 && forward[det] != reverse[det] )
        Correlation discrepancy at:
```

```
            band, SCA, detector number, forward[det], reverse[det]
           status = 1
            if img_sel[det] < 0 && forward[det] > -90
                if curdet[det]%2 == 0 && forward[det] == 0
               img_sel[det] = 0
            if curdet[det]%2 == 0 && forward[det] == -2
                img_sel[det] = 1
            if swir == 1 && curdet[det]%2 == 0 && forward[det] == -4
                img_sel[det] = 2
            if curdet[det]%2 == 1 && forward[det] == 2
               img_sel[det] = 0
            if curdet[det]%2 == 1 && forward[det] == 4
               img_sel[det] = 1
           if swir == 1 && curdet[det]%2 == 1 && forward[det] == 6
              img_sel[det] = 2

       if img_sel[det] < 0
           Print: Error assigning image selection at:  band, SCA, det
        if status == 0 && cpf_sel[det] >= 0 && img_sel[det] >= 0 &&  cpf_sel[det] !=
img_sel[det]
            Print: Detector Selection Error:  band, SCA, det+1, cpf_sel[det],
img_sel[det]
              if total == 0
                 Output to file: column header
              Output to file : band, SCA, det+1, cpf_sel[det], img_sel[det]
          count++
          total++

    1.6) if status != 0
              Output to file: Suspect correlation: Band, SCA

    1.7) Print: band, SCA, det+1, cpf_sel[det], img_sel[det]
2)  Finished with getmap.
```

Results Aggregation and Reporting

Implementation for the DMV algorithm should contain methods for accessing the database for all the individual scenes trended for the detmap and getmap steps in a manner that allows for a assessment of the state of the detector over a given time range.

4.2.14.7 Trending Output

There should be two types of trending that occurs with this algorithm.

1) The first would be a table with the scene ID, or information along those lines such as path/row/acquisition date, and whether the outcome is either:
   a) Bad scene. One for which detector misalignment was found.

b) Good scene. One for which the correlation and offset calculations were successful and no detector misalignment was found.
c) Unknown scene. That for which the correlation or offset calculations were inconclusive and it could not be determined if the scene has any detector misalignment.

The second should be if a scene was classified as 1a) above then what is wrong with that scene. The band, SCA, and detectors that were found to be misaligned.

4.2.14.8 Notes

Some additional background assumptions and notes include:

1. The current output files, those of the detmap and getmap prototype executables, do not contain the standard IAS output headers (as of 02/21/2018). They should be added to the output files generated. For the detmap output files there should also be column headers indicating SCA, detector, measured offset, correlation peak calculated, left-fill, right-fill, nominal fill, current detector delay (that of current detector), next detector delay (that of current +1 detector).
2. The process fails to work well, or really at all, with the cirrus band. Default processing should be to omit processing the cirrus band.
3. Any kind of notification about a possible detector map issue should be found out through a CalVal member querying the database.
4. What should be done if a detector misalignment issue has been found is outside the scope of this description.

4.3 TIRS Geometry Algorithms

4.3.1 TIRS Line-of-Sight Model Creation

4.3.1.1 Background/Introduction

The LOS model creation algorithm gathers the ancillary data and calibration parameters required to support geometric processing of the input TIRS/TIRS-2 image dataset; validates the image time codes; extracts, validates, and preprocesses the TIRS SSM telemetry contained in the ancillary data stream; extracts the corresponding ephemeris and attitude data from the ancillary data stream; performs the necessary coordinate transformations; and stores the results in a geometric model structure for subsequent use by other geometric algorithms. The TIRS LOS model creation algorithm is derived from the OLI model creation algorithm. Its implementation will be very similar to the corresponding OLI application and will draw on the same spacecraft model, math, and utility libraries. Note that the ephemeris and attitude preprocessing logic common to both sensors is performed by the ancillary data preprocessing algorithm (4.1.4) to isolate the bulk of the geometric processing logic from the details of the incoming ancillary data stream. New attitude data processing logic has also been added to separate the high- and low-frequency attitude effects to allow the image resampling process to better correct for jitter at frequencies above the original 10 Hz algorithm design limit without requiring an unreasonably dense resampling grid.

4.3.1.2 Dependencies

The TIRS LOS Model Creation algorithm assumes that the Ancillary Data Preprocessing algorithm (4.1.4) has been executed to accomplish the following:

- Validated ephemeris data for the full imaging interval have been generated
- Validated attitude data for the full imaging interval have been generated
- The ancillary data have been scaled to engineering units

The Ancillary Data Preprocessing algorithm will generate preprocessed smoothed and cleaned ephemeris and attitude data streams. This provides a standard validated input for subsequent LOS model generation.

4.3.1.3 Inputs

The TIRS LOS Model Creation algorithm uses the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data; including dataset IDs to provide unique identifiers for data trending).

Algorithm Inputs
ODL File (implementation)
Acquisition Type (Earth, Lunar, Stellar) (optional, defaults to Earth)
CPF File Name
Preprocessed Ancillary Data Input File Name
L0R/L1R File Name
WRS Path/Row (stored in model and used for trending)
Trending On/Off Switch
Geometric work order common characterization ID (for trending)
Work Order ID (for trending)
Optional Precision Model Input Parameters (see note 6)
Input Precision Model Reference Time (optional)
Input Precision Ephemeris Correction Order (optional)
Input Precision X Correction Parameters (optional)
Input Precision Y Correction Parameters (optional)
Input Precision Z Correction Parameters (optional)
Input Precision Attitude Correction Order (optional)
Input Precision Roll Correction Parameters (optional)
Input Precision Pitch Correction Parameters (optional)
Input Precision Yaw Correction Parameters (optional)
CPF Contents
WGS84 Earth ellipsoid parameters
Earth orientation parameters (UT1UTC, pole wander, leap seconds) (see note 1)
Earth rotation velocity
Speed of light
TIRS to ACS reference alignment matrix/quaternion
Spacecraft center of mass (CM) to TIRS offset in ACS reference frame
High frequency attitude data cutoff frequency (Hz)
Scene select mirror calibration parameters (see Table 4-34 below)

Algorithm Inputs
Focal plane model parameters (Legendre coefficients)
TIRS detector delay table (including whole pixel deselect offsets) (see note 10)
Nominal L0R fill (per band/SCA)
Nominal TIRS frame time nominal_frame_time (14.2857143 msec)
Nominal TIRS integration time
Image time code outlier thresholds delta_time_tolerance (DTIME_TOL) and time_outlier_tolerance (OUTLIER_TOL) (see note 3)
SSM encoder outlier threshold (see note 7)
Preprocessed Ancillary Data Contents
Attitude Data
Attitude data UTC epoch: Year, Day of Year, Seconds of Day
Time from epoch (one per sample, nominally 50 Hz)
ECI quaternion (vector: q1, q2, q3, scalar: q4) (one per sample)
ECEF quaternion (one per sample)
Body rate estimate (roll, pitch, yaw rate) (one per sample)
Roll, pitch, yaw estimate (one per sample)
Ephemeris Data
Ephemeris data UTC epoch: Year, Day of Year, Seconds of Day
Time from epoch (one per sample, nominally 1 Hz)
ECI position estimate (X, Y, Z) (one set per sample)
ECI velocity estimate (Vx, Vy, Vz) (one set per sample)
ECEF position estimate (X, Y, Z) (one set per sample)
ECEF velocity estimate (Vx, Vy, Vz) (one set per sample)
L0R/L1R Data Contents
Image Time Codes (one per line)
Integration Time
Scene Select Mirror Telemetry Packets (from Ancillary Data, see Table 4-35 below)
TIRS Ancillary Data Time Code (one per 1 Hz frame)
Mirror Encoder Readout (24 bits, in counts) (20 samples per 1 Hz frame) (see note 9)
Detector Alignment Fill Table (see note 2)

4.3.1.4 Outputs

TIRS LOS Model (additional detail is provided in Table 4-36 below)
WGS84 Earth ellipsoid parameters
Earth Orientation Parameters (for current day) from CPF
Earth rotation velocity
Speed of light
TIRS to ACS reference alignment matrix/quaternion
Spacecraft center of mass to TIRS offset in ACS reference frame
SSM model parameters (Telescope alignment matrix and preprocessed SSM angles)
Focal plane model parameters (Legendre coefs)
Detector delay table (including whole pixel deselect offsets)
Nominal detector alignment fill table (from CPF)
L0R detector alignment fill table (from L0R)
ECI J2000 spacecraft ephemeris model (original and corrected)
ECEF spacecraft ephemeris model (original and corrected)
Spacecraft attitude model (time, roll, pitch, yaw) (orig and corr) (see note 4)
High frequency attitude perturbations (roll, pitch, yaw) per image line (jitter table)

TIRS LOS Model (additional detail is provided in Table 4-36 below)
Image time codes (see note 5) (in seconds)
Integration Time (in seconds)
Sample Time (in seconds)
WRS Path/Row
Model Trending Data
WRS Path/Row
Acquisition Date/Time
Geometric work order common characterization ID
Work Order ID
Image start UTC time (year, day of year, seconds of day)
Computed image frame time (in seconds)
Number of image lines
Number of out of limit image time codes
Number of out of limit SSM time codes
Number of out of limit SSM encoder measurements

4.3.1.5 Options

Trending On/Off Switch
Optional precision model input parameters can be used to force model corrections.

4.3.1.6 Procedure

The TIRS LOS model is stored as a structure and is created from information contained in the Level 0R or Level 1R image data, the CPF, and the Ancillary data. The model is subsequently used along with the CPF to create a resampling grid. Data present in the model structure includes satellite position, velocity, and attitude, LOS angles, timing references, scene select mirror position, precision correction information (if any), and the software version. The TIRS LOS model is also used in several characterization and calibration routines for mapping input line/sample locations to geographic latitude/longitude.

The TIRS LOS model may be thought of in two parts, an instrument model that provides a line-of-sight vector for each TIRS detector (and, hence, each image line/sample), and a spacecraft model that provides spacecraft ephemeris (position and velocity) and attitude as a function of time. These models are linked by the image time stamps that allow each Level 0R or Level 1R image sample to be associated with a time of observation. The spacecraft portion of the model is common to the OLI LOS model.

Instrument Model
Figure 4-57 shows the arrangement of the bands and SCAs on the TIRS focal plane. The model treats every band of every SCA independently. This is done by defining a set of 3rd order Legendre polynomials (see note #11) for each band of each SCA. Unlike the OLI, the TIRS detectors are arranged in a two-dimensional array with two rows of that array being downlinked for each spectral band. One of the downlinked rows is primary and the other, redundant row is only used to replace bad pixels in the primary row. The TIRS LOS Legendre polynomials represent a theoretical “nominal” set of detectors that are best-fit to the primary row of detectors. This approach treats any replaced detectors

as though they were aligned with the primary detectors for purposes of sensor LOS generation. This is a simplification of the OLI approach, which also must account for even/odd detector stagger. In the TIRS case, this stagger is effectively set to zero. This approach explicitly models any offsets caused by detector replacement, and the subpixel deviations of each detector from its nominal (Legendre best fit) location, for correction during image resampling. This leads to three detector types: nominal, actual, and exact. A nominal detector is calculated from the Legendre polynomials. An actual detector corrects the nominal detector location for the nominal (whole pixel) detector select offsets. Like the OLI, since individual detectors may be deselected/replaced, these offsets are detector dependent. An exact detector has the actual correction applied but also includes the specific individual (subpixel) detector offsets. The Legendre polynomials and a table of detector offset values are stored in the CPF.

There is a slight angular difference between the line of sight vectors or angles associated with the primary and any replaced detectors. If the nominal LOS, generated using the 3rd order Legendre model, is $\psi_{nominal}$, the look angles for the actual and exact detectors are as follows:

$$\psi x\_actual = \psi x\_nominal + round(detector\_shift\_x) * IFOV$$
$$\psi y\_actual = \psi y\_nominal + round(detector\_shift\_y) * IFOV$$

$$\psi x\_exact = \psi x\_nominal + detector\_shift\_x * IFOV$$
$$\psi y\_exact = \psi y\_nominal + detector\_shift\_y * IFOV$$

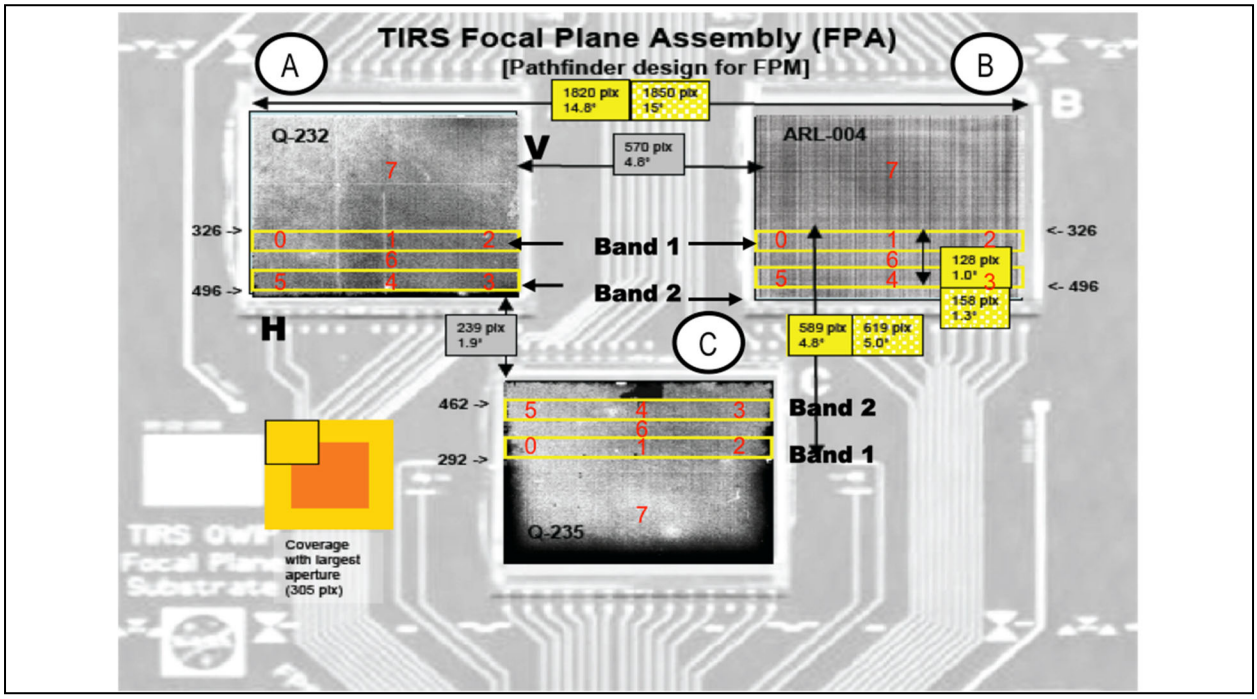

Figure 4-57. TIRS Focal Plane Layout

The detector_shift_x and detector_shift_y values are the detector-specific offsets stored in the CPF detector delay tables. These offsets include both the whole-pixel deselect/replacement offsets and the fractional-pixel detector placement effects, and must be rounded to extract the integer portion. Note that the integer portion of the detector_shift_y value is always zero since the deselect effects are applicable only in the X direction. Also note that the integer portion of the detector_shift_x values will also all be zero in the event that no primary detectors are deselected.

The nominal LOS is used in most line-of-sight projection applications. The actual LOS is used in conjunction with the actual image time (see below) to model the errors introduced by trading time (sample delay) for space (detector offset) for purposes of correcting the nominal LOS model. The exact LOS is generally used only for data simulation and other analytical purposes rather than in the geometric correction model, as the subpixel portion of the detector delay is applied directly in the image resampler rather than being included in the LOS model.

Sample Timing

The TIRS provides a timestamp with each image line collected. These timestamps make it possible to relate the image samples (pixels) to the corresponding spacecraft ephemeris and attitude data. The TIRS timestamps are contained in a line header that precedes each image line. The TIRS line header contents are shown in Figure 4-58. Several items in this figure are worthy of particular note. First, the timestamp associated with a data frame is recorded at the beginning of the detector integration period. Second, the line header includes the integration time (TIRS does not use a separate frame header) and identifies the detector rows selected for downlink for each band and SCA. This includes the dark band, which is not included in the TIRS geometric model. Line header fields other than the time code should be static within an imaging interval.

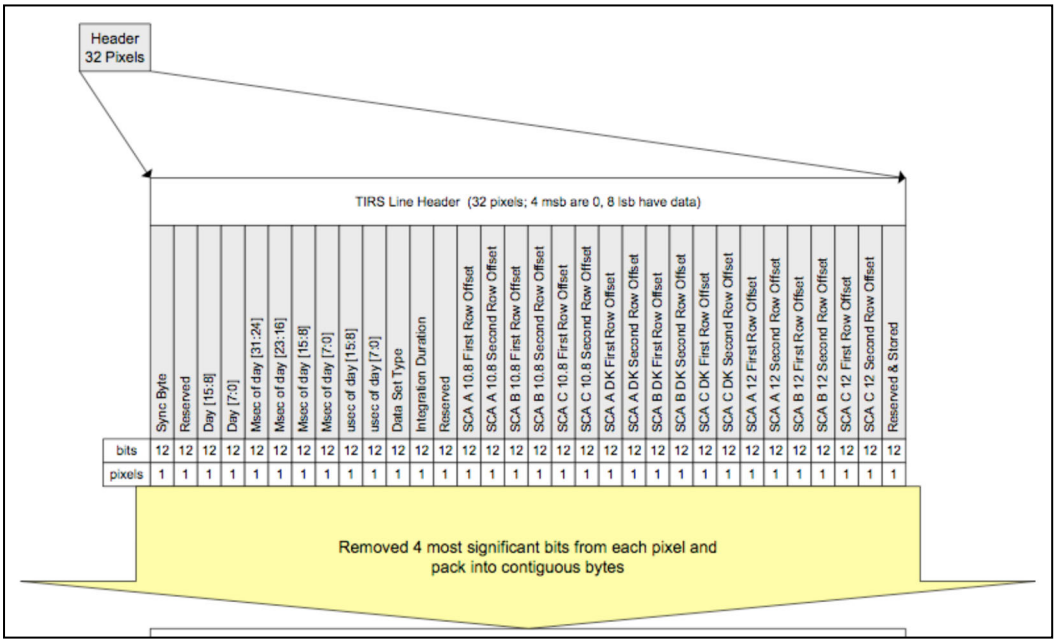

Figure 4-58. TIRS Line Header Contents

Note that having the time code define the start of detector integration is different than the OLI where the time code represents the end of integration. This has the effect of making the integration time correction a positive adjustment to the pixel time for TIRS rather than a negative adjustment, as is the case for the OLI.

Also note that the TIRS line header is composed of 12-bit data words, like the other TIRS “pixels,” where the most significant 4 bits are all zero. The assumption here is that Level 0 processing will treat the line header words the same way that it handles TIRS pixel data and repackage the 12-bit fields into 16-bit data words. If instead, Level 0 processing strips off the extra 4 bits from the line header fields, then the line header preprocessing step mentioned below, will not be necessary.

One further complication to the problem of assigning times to image samples is the fact that the Level 0R/1R input imagery may include fill pixels inserted to achieve nominal primary and replaced detector alignment. This fill insertion allows the geometrically unprocessed 0R/1R imagery to be viewed as a spatially contiguous image without detector misalignments. The amount of detector alignment fill present will be indicated in the L0R/L1R image data (this is the purpose of the detector alignment fill table input noted above) so that the association of image samples with their corresponding time stamps can be adjusted accordingly. The assumption here is that, like OLI data, image fill will not be used to achieve nominal SCA or band alignment for TIRS data.

Due to the potential for deselected/replaced detectors, the nominal and actual times associated with a given pixel may not be the same. The actual time reflects the time that the current detector was actually sampled whereas the nominal time reflects the time at which the idealized detector represented by the TIRS LOS model would have been sampled.

If the current position within the image is given as a line and sample location, the two different “types” of times for TIRS pixels are calculated by:

```
l0r_fill_pixels = round(detector_shift_x) + nominal_fill_pixels
time_index = line_number - l0r_fill_pixels
if ( time_index < 0 ) time_index = 0
if (time_index > (num_time_stamps - 1)) time_index = num_time_stamps - 1

actual_time = line_time_stamp[time_index] + integration_time/2
                    + (line_number - l0r_fill_pixels - time_index) * TIRS_sample_time

nominal_time = actual_time + round(detector_shift_x) * TIRS_sample_time
```

where:

- line_number is the zero-referenced TIRS image line number (N).
- nominal_fill_pixels is the amount of detector alignment fill to be inserted at the beginning of pixel columns that correspond to nominal detectors; that is,

those detectors with a delay value of zero that are the basis for the Legendre polynomial LOS model. This value comes from the CPF and will be zero if there are no bad detectors to replace.

- l0r_fill_pixels is the total amount of detector alignment fill to be inserted at the beginning of the pixel column associated with the current detector. It includes both the nominal_fill_pixels and the detector-specific delay fill required to align deselected/replaced detectors.
- num_time_stamps is the total number of time codes (image lines) in the image. It is tested to ensure that time_index, the line_time_stamp index, does not go out of bounds.
- detector_shift_x is the amount of detector offset for the current detector from the TIRS LOS model detector delay table. It is rounded to the nearest integer pixel because time offsets can only occur in whole line increments.

The detector_shift_x parameter is the detector-specific along-track offset as recorded in the CPF and subsequently stored in the LOS model detector delay table. It is rounded to the nearest integer so as to include the effects of even/odd detector stagger and detector deselect, but not the detector-specific subpixel offsets. The L0R/L1R data can be accessed by SCA making the association of sample number with detector index more straightforward. Note that, like OLI, the TIRS Level 0R data organizes the image samples from all 3 TIRS SCAs so that the samples are numbered left-to-right for all SCAs. This convention is also followed in the CPF detector offset tables. There are 640 samples per SCA for each spectral band.

Note that when fill is used to align replaced detectors the spatial difference between the nominal and actual look vectors is approximately compensated by the time difference between $t_{nominal}$ and t_{actual}.

TIRS Scene Select Mirror Model

The TIRS SSM redirects the lines-of-sight from the TIRS telescope, which is oriented with its optical axis nominally in the +X direction, toward either: 1) the nadir Earth view; or 2) the space view port; or 3) the internal black body. It is the Earth view case that is of interest to the geometric processing models. Figure 4-59 shows the SSM and TIRS telescope in relationship to the TIRS coordinate system, in which the telescope optical axis defines the +X axis. The TIRS coordinate system is nominally aligned with the spacecraft coordinate system (+X toward the direction of flight, +Z toward nadir, +Y completing a right-handed coordinate system).

The SSM angle, γ, is defined as the angle between the SSM normal and the SSM axis of rotation. This angle is nominally $\pi/4$ radians (45 degrees). The SSM angle is a parameter of the SSM system that is stored in the CPF.

Define a scene select mirror coordinate system, nominally parallel to the TIRS coordinate system, with the +X axis parallel to the SSM axis of rotation ($\underline{\mathbf{X}}_{ssm}$), the +Y axis in the direction of the cross product of the mirror normal vector $\underline{\mathbf{n}}_{\mathbf{ssm}}$ and $\underline{\mathbf{X}}_{ssm}$, and the +Z axis completing a right handed coordinate system.

The mirror axis of rotation and mirror normal are as follows:

$$\underline{\mathbf{X}}_{ssm} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad \underline{\mathbf{n}}_{ssm} = \begin{bmatrix} -\cos(\gamma) \\ 0 \\ \sin(\gamma) \end{bmatrix}$$

To include the effect of SSM rotation about its axis, rotate $\underline{\mathbf{n}}_{ssm}$ about $\underline{\mathbf{X}}_{ssm}$ by an angle $(\theta - \theta_0)$ where θ_0 is the SSM encoder angle at the nominal nadir pointing angle and θ is the actual SSM encoder angle reported in the TIRS ancillary data. The mirror normal as a function of θ is:

$$\underline{\mathbf{n}}_{ssm}(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta - \theta_0) & -\sin(\theta - \theta_0) \\ 0 & \sin(\theta - \theta_0) & \cos(\theta - \theta_0) \end{bmatrix} \begin{bmatrix} -\cos(\gamma) \\ 0 \\ \sin(\gamma) \end{bmatrix} = \begin{bmatrix} -\cos(\gamma) \\ -\sin(\gamma)\sin(\theta - \theta_0) \\ \sin(\gamma)\cos(\theta - \theta_0) \end{bmatrix}$$

The nominal nadir pointing angle θ_0 is a SSM calibration parameter stored in the CPF. The measured encoder angle as a function of time, $\theta(t)$, is reported in the 1 Hz TIRS ancillary data stream, with 20 samples provided in each 1 Hz TIRS ancillary data packet. Any time delay between the actual encoder sample time(s) and the corresponding ancillary data packet time code must be accounted for, so this time delay, Δt, is included as a parameter in the CPF.

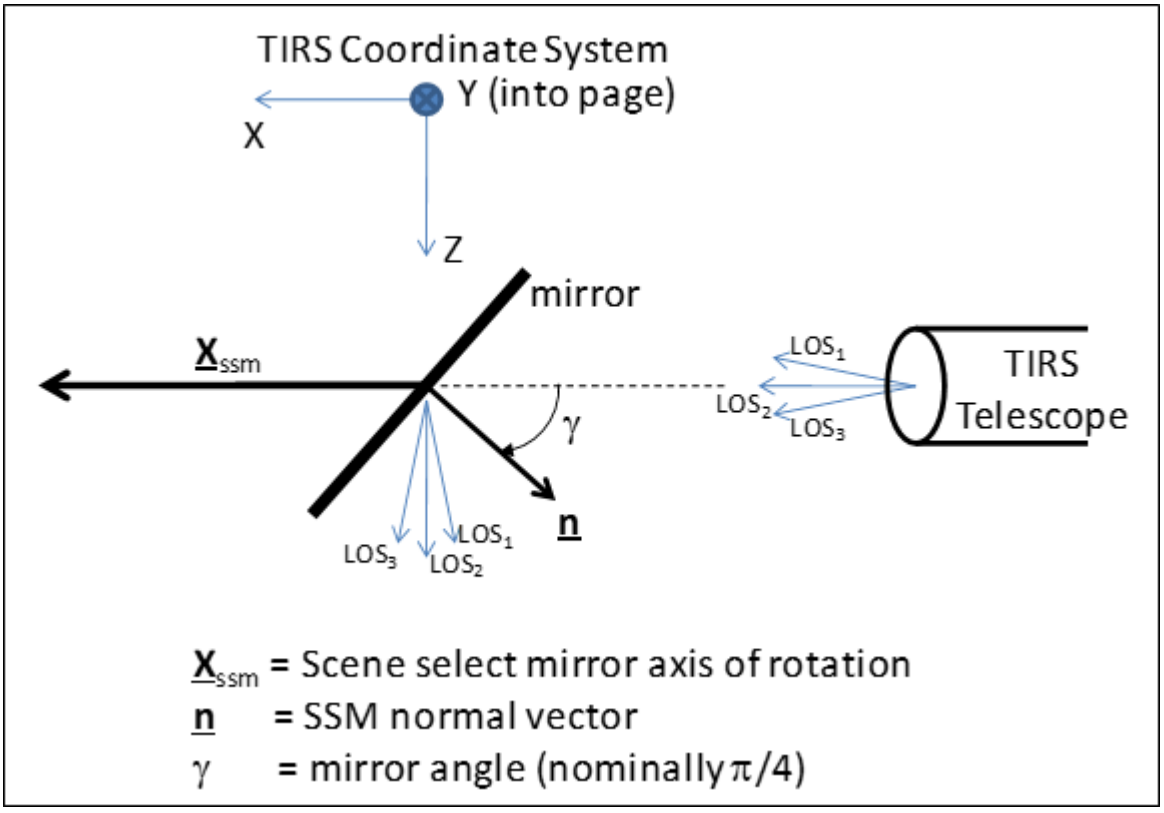

Figure 4-59. Scene Select Mirror Line-of-Sight Redirection

In the TIRS telescope coordinate system, the LOS emerging from the telescope, $\underline{\mathbf{l}}_{tele}$, is:

$$\underline{\mathbf{l}}_{tele} = \begin{bmatrix} 1 \\ \tan(\delta_{XT}) \\ \tan(\delta_{AT}) \end{bmatrix}$$

where:

δ_{XT} is the across-track LOS angle (from the Legendre polynomial model)
δ_{AT} is the along-track LOS angle (from the Legendre polynomial model)

To account for misalignments between the SSM and the TIRS telescope, rotate $\underline{\mathbf{l}}_{tele}$ about the X axis by an angle Δr, about the Y axis by an angle Δp, and about the Z axis by an angle Δy:

$$\boldsymbol{l}_{SSM} = \boldsymbol{M}(\Delta r, \Delta p, \Delta y)\ \boldsymbol{l}_{tele}$$

Where:

$$\mathbf{M}(\Delta r, \Delta p, \Delta y) = \begin{bmatrix} \cos\Delta y & \sin\Delta y & 0 \\ -\sin\Delta y & \cos\Delta y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\Delta p & 0 & -\sin\Delta p \\ 0 & 1 & 0 \\ \sin\Delta p & 0 & \cos\Delta p \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\Delta r & \sin\Delta r \\ 0 & -\sin\Delta r & \cos\Delta r \end{bmatrix}$$

$$= \begin{bmatrix} \cos\Delta p\cos\Delta y & \sin\Delta r\sin\Delta p\cos\Delta y + \cos\Delta r\sin\Delta y & \sin\Delta r\sin\Delta y - \cos\Delta r\sin\Delta p\cos\Delta y \\ -\cos\Delta p\sin\Delta y & \cos\Delta r\cos\Delta y - \sin\Delta r\sin\Delta p\sin\Delta y & \cos\Delta r\sin\Delta p\sin\Delta y + \sin\Delta r\cos\Delta y \\ \sin\Delta p & -\sin\Delta r\cos\Delta p & \cos\Delta r\cos\Delta p \end{bmatrix}$$

The telescope misalignment angles, Δr, Δp, and Δy, are calibration parameters stored in the CPF.

The LOS vector is reflected off the SSM by multiplying it by the matrix $\mathbf{P}(\theta)$:

$$\mathbf{P}(\theta) = \mathbf{I} - 2\ \underline{\mathbf{n}}_{SSM}(\theta)\ \underline{\mathbf{n}}^{T}_{SSM}(\theta)$$

And the resulting reflected LOS is:

$$\underline{\mathbf{l}}_{TIRS}(\theta) = \mathbf{P}(\theta)\ \underline{\mathbf{l}}_{SSM} = [\mathbf{I} - 2\ \underline{\mathbf{n}}_{SSM}(\theta)\ \underline{\mathbf{n}}^{T}_{SSM}(\theta)]\ \mathbf{M}(\Delta r,\Delta p,\Delta y)\ \underline{\mathbf{l}}_{tele}$$

Define $= \theta - \theta_0$, and $\gamma = \pi/4 + \Delta\gamma$, where $\Delta\gamma$ is the departure from the ideal mirror angle.

The corresponding reflection matrix, **P**, becomes:

$$\mathbf{P}(\hat{\theta}, \Delta\gamma) = \begin{bmatrix} \sin 2\Delta\gamma & -\cos 2\Delta\gamma\sin\hat{\theta} & \cos 2\Delta\gamma\cos\hat{\theta} \\ -\cos 2\Delta\gamma\sin\hat{\theta} & \cos^2\hat{\theta} - \sin 2\Delta\gamma\sin^2\hat{\theta} & \sin\hat{\theta}\cos\hat{\theta}(1+\sin 2\Delta\gamma) \\ \cos 2\Delta\gamma\cos\hat{\theta} & \sin\hat{\theta}\cos\hat{\theta}(1+\sin 2\Delta\gamma) & \sin^2\hat{\theta} - \sin 2\Delta\gamma\cos^2\hat{\theta} \end{bmatrix}$$

For an ideal SSM, $\Delta\gamma = 0$, so the ideal reflection matrix, $\mathbf{P}_0(\hat{\theta})$, becomes:

$$\mathbf{P}_0(\hat{\theta}) = \begin{bmatrix} 0 & -sin\hat{\theta} & cos\hat{\theta} \\ -sin\hat{\theta} & cos^2\hat{\theta} & sin\hat{\theta}\, cos\hat{\theta} \\ cos\hat{\theta} & sin\hat{\theta}\, cos\hat{\theta} & sin^2\hat{\theta} \end{bmatrix}$$

Which for nadir pointing ($\hat{\theta}$=0) reduces to:

$$\mathbf{P}_0(0) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

For a perfectly aligned SSM, $\Delta r = \Delta p = \Delta y = 0$, so:

$$\underline{\mathbf{l}}_{TIRS}(0) = \mathbf{P}_0(0)\,\underline{\mathbf{l}}_{tele} = \begin{bmatrix} \tan(\delta_{AT}) \\ \tan(\delta_{XT}) \\ 1 \end{bmatrix}$$

Note that this matches the nadir-pointing LOS formulation used for the OLI. To minimize the differences between the standard nadir-pointing (OLI) LOS model and the SSM reflected (TIRS) LOS model, we formulate the SSM effect as a rotation applied to a nadir-pointing LOS vector as follows:

Noting that: $\mathbf{P}_0(0)\,\mathbf{P}_0(0) = \mathbf{I}$

$\underline{\mathbf{l}}_{TIRS}(\theta) = [\mathbf{I} - 2\,\underline{\mathbf{n}}_{SSM}(\theta)\,\underline{\mathbf{n}}^T_{SSM}(\theta)]\,\mathbf{P}_0(0)\,\mathbf{P}_0(0)\,\mathbf{M}(\Delta r,\Delta p,\Delta y)\,\mathbf{P}_0(0)\,\mathbf{P}_0(0)\,\underline{\mathbf{l}}_{tele}$

$\underline{\mathbf{l}}_{TIRS}(\theta) = [\mathbf{P}_0(\hat{\theta},\Delta\gamma)\,\mathbf{P}_0(0)]\,[\mathbf{P}_0(0)\,\mathbf{M}(\Delta r,\Delta p,\Delta y)\,\mathbf{P}_0(0)]\,\underline{\mathbf{l}}_{TIRS}(0)$

The TIRS LOS, $\underline{\mathbf{l}}_{TIRS}(\theta)$, can thus be written as the ideal nadir-pointed LOS , $\underline{\mathbf{l}}_{TIRS}(0)$, rotated by the telescope alignment matrix, $\mathbf{M'}(\Delta r,\Delta p,\Delta y)$ and reflected by the SSM matrix, $\mathbf{P'}(\hat{\theta},\Delta\gamma)$:

$\underline{\mathbf{l}}_{TIRS}(\theta) = \mathbf{P'}(\hat{\theta},\Delta\gamma)\,\mathbf{M'}(\Delta r,\Delta p,\Delta y)\,\underline{\mathbf{l}}_{TIRS}(0)$

where:

$$\mathbf{P'}\left(\hat{\theta},\Delta\gamma\right) = \begin{bmatrix} \cos2\Delta\gamma\cos\hat{\theta} & -\cos2\Delta\gamma\sin\hat{\theta} & \sin2\Delta\gamma \\ \sin\hat{\theta}\cos\hat{\theta}\left(1+\sin2\Delta\gamma\right) & \cos^2\hat{\theta}-\sin2\Delta\gamma\sin^2\hat{\theta} & -\cos2\Delta\gamma\sin\hat{\theta} \\ \sin^2\hat{\theta}-\sin2\Delta\gamma\cos^2\hat{\theta} & \sin\hat{\theta}\cos\hat{\theta}\left(1+\sin2\Delta\gamma\right) & \cos2\Delta\gamma\cos\hat{\theta} \end{bmatrix}$$

$$\mathbf{M}'(\Delta r, \Delta p, \Delta y) =$$
$$\begin{bmatrix} \cos\Delta r \cos\Delta p & -\sin\Delta r \cos\Delta p & \sin\Delta p \\ \cos\Delta r \sin\Delta p \sin\Delta y + \sin\Delta r \cos\Delta y & \cos\Delta r \cos\Delta y - \sin\Delta r \sin\Delta p \sin\Delta y & -\cos\Delta p \sin\Delta y \\ \sin\Delta r \sin\Delta y - \cos\Delta r \sin\Delta p \cos\Delta y & \sin\Delta r \sin\Delta p \cos\Delta y + \cos\Delta r \sin\Delta y & \cos\Delta p \cos\Delta y \end{bmatrix}$$

For an ideal SSM, the product of the reflection and alignment rotation matrices, $\mathbf{P'}_0(\hat{\theta},0)$ $\mathbf{M'}(0,0,0)$, reduces to:

$\mathbf{M}_0(\hat{\theta}) = \mathbf{P'}_0(\hat{\theta},0)\ \mathbf{M'}(0,0,0) = \begin{bmatrix} cos\hat{\theta} & -sin\hat{\theta} & 0 \\ sin\hat{\theta}\ cos\hat{\theta} & cos^2\hat{\theta} & -sin\hat{\theta} \\ sin^2\hat{\theta} & sin\hat{\theta}\ cos\hat{\theta} & cos\hat{\theta} \end{bmatrix}$, and $\mathbf{M}_0(0) = \mathbf{I}$

Noting that $\hat{\theta}$, $\Delta\gamma$, Δr, Δp, and Δy are all close to zero, it can be shown that the $\Delta\gamma$ mirror angle offset is approximately equivalent to a pitch misalignment of $2\Delta\gamma$. Using this approximation we can write the TIRS LOS transformation equation as the product of a reflection matrix, that is a function of only the mirror rotation angle θ, and a static telescope alignment matrix:

$$\underline{\mathbf{l}}_{TIRS}(\theta) \approx \mathbf{M}_0(\hat{\theta})\ \mathbf{M'}(\Delta r, \Delta p + 2\Delta\gamma, \Delta y)\ \underline{\mathbf{l}}_{TIRS}(0)$$

In practice, the SSM reflection matrix $\mathbf{M}_0(\hat{\theta})$ will be close to **I** so it will be difficult to distinguish telescope misalignments, modeled by **M'**, from TIRS instrument misalignments, and corrections to any prelaunch telescope alignment parameters will be absorbed by the TIRS alignment angles estimated on-orbit. For this reason we do not anticipate performing on-orbit calibration for the SSM parameters. As a historical note, such an on-orbit calibration was performed for the TIRS instrument when image data were acquired with the SSM (accidentally) pointed off-nadir during the SSM anomaly investigation. The TIRS SSM calibration parameters are summarized in Table 4-34. Note that since the primary (side A) and redundant (side B) SSM encoders are not perfectly aligned, the encoder values that correspond to nadir pointing will not be exactly the same. Thus, the nominal nadir encoder angle, θ_0, will be equal to either θ_A or θ_B depending upon the mirror side/encoder in use. Similarly, the commanded nadir encoder counts will be either N_A or N_B depending upon the encoder side in use. The mirror side in use will be indicated in the TIRS ancillary data. For L8 this is reflected in the mirror control electronics (MCE) side that is enable (A or B). For L9, the encoders are cross-strapped so that either encoder can be read by either MCE side, and there is a separate telemetry field (T_MC_ENC_SEL) that identifies which encoder is active, independent of the MCE side.

Note that for TIRS-2 the nadir pointing angle and the commanded nadir position (in counts) are both provided. This is due to the fact that these may not represent exactly the same position. The nadir angle ($\theta_{A/B}$) is the encoder position at which the SSM is pointing in the direction defined as the zero reference for the TIRS calibrated pointing model. The commanded mirror position ($N_{A/B}$) may be offset slightly (80 arc-seconds or less) from this position, and the offset may be different for the A and B encoders. This

offset is introduced to position the active encoder at a location where the encoder analog signal levels are such that they minimize the likelihood of inducing a loss-of-index anomaly. The commanded position provides a reference value to correct the corrupted high order encoder telemetry bits should a loss-of-index anomaly occur.

Parameter	Symbol	Source
Mirror Angle	γ	Fixed constant
Mirror Angle Deviation	$\Delta\gamma$	Prelaunch characterization
Nadir Pointing Angle / Encoder Origin (Side A)	θ_A	Defined value – establishes reference for TIRS alignment calibration
Nadir Pointing Angle / Encoder Origin (Side B)	θ_B	Defined value – establishes reference for TIRS alignment calibration
Commanded Nadir Counts Side A	N_A	Flight operations value, may be changed on-orbit to align TIRS-2 with OLI-2
Commanded Nadir Counts Side B	N_B	Flight operations value, may be changed on-orbit to align TIRS-2 with OLI-2
Telescope Roll Offset	Δr	Prelaunch characterization
Telescope Pitch Offset	Δp	Prelaunch characterization
Telescope Yaw Offset	Δy	Prelaunch characterization
Encoder Time Offset	Δt	Prelaunch characterization

Table 4-34. TIRS Scene Select Mirror Model Calibration Parameters

Spacecraft Model
The spacecraft ephemeris and attitude models are constructed from the input preprocessed ancillary data by extracting the ancillary data that span the current image. Both ECI and ECEF versions of the ephemeris data are retained in the model structure to avoid the need to repeatedly invoke the ECI/ECEF coordinate system conversion. The ALIAS heritage roll-pitch-yaw representation of the attitude model is retained in the model structure though a quaternion representation may be used in a future algorithm revision (see note 4).

Prepare TIRS LOS Model Sub-Algorithm
This function gathers the information from the preprocessed ancillary data and the Level 0R/1R TIRS image and ancillary data needed to process model data and run the TIRS LOS model. Though it has the same overall purpose and function as the OLI oli_run_model unit, additional logic is required to handle the TIRS SSM telemetry information. The spacecraft (preprocessed ancillary data) sections are the same as the OLI model.

The main steps are as follows:

1. Load the image time codes and convert to seconds since spacecraft epoch.
2. Determine the image time window.
3. Validate/smooth the image time codes.
4. Extract the integration time from the Level 0R/1R image line header data.
5. Extract and preprocess the SSM telemetry from the Level 0R ancillary data.

6. Extract the associated ephemeris and attitude data from the preprocessed ancillary data stream.
7. Preprocess the input attitude data into a low-frequency stream, used for basic geometric modeling, and a high-frequency stream, used as a fine correction in the image resampler. This preprocessing was added to improve the ability of the geometric correction system to compensate for jitter disturbance frequencies above 10 Hz.

The input preprocessed ancillary data are stored in an HDF file. The attitude and ephemeris ancillary data streams each have an epoch time identifying the UTC date/time reference. Within these data streams, each attitude or ephemeris observation in the HDF file has a corresponding time offset relative to the epoch. This incoming ancillary data stream spans the entire imaging interval containing the image data represented in the Level 0R/1R input data. In creating the model we identify and extract the ancillary data sequence required to process the current image data.

The input Level 0R/1R image data are also packaged in HDF files that include the image samples for each band and SCA and the time codes assigned to each image line by the TIRS instrument. As shown in Figure 4-58 above and Figure 4-60 below, these spacecraft time codes are provided by the TIRS in CCSDS T-Field format which includes days since epoch (16-bit integer), milliseconds of day (32-bit integer) and microseconds of millisecond (16-bit integer) fields:

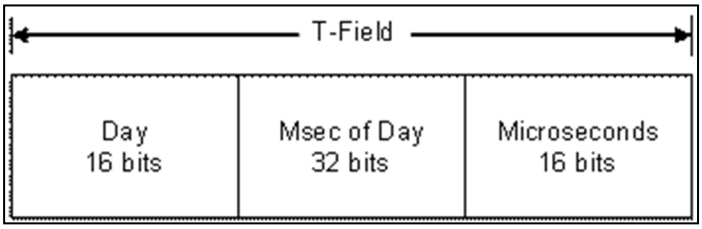

Figure 4-60. TIRS Time Code Format

The baseline algorithm assumes that Level 0 processing will preserve these time codes in their original form, as days, milliseconds, and microseconds since the spacecraft epoch. Since they are derived from the spacecraft clock, the image time codes will be based on the same epoch used by the ancillary data (e.g., TAI seconds from J2000). Like all fields in the TIRS line header, the time codes are packaged in 12 bit fields with only the low order 8 bits containing valid data (see Figure 4-58). Any initial line header preprocessing steps necessary to extract the 8 valid data bits from each line header data word are assumed to have been performed as part of Level 0Rp generation.

Process Image Time Codes
The image time codes are loaded from the input HDF Level 0R/1R dataset. Deselected/replaced detector alignment fill will be inserted into the Level 0R/1R imagery as described above, if necessary, so the image lines each contain samples collected at times that may be offset from the time specified by the corresponding time code. The relationship between these time codes, the TIRS integration time, and the pixel center

times has already been described above. The assumption here is that the L0Rp data will contain one time code per image line, excluding any fill lines, or a nominal 2801 time codes per scene. The image files themselves may be up to 30 lines longer to accommodate the redundant row deselect/replacement detector-alignment fill pixels. Simulated scenes may also be longer to provide the additional scene-to-scene overlap needed to support interval stitching.

1. Convert the time code to seconds from spacecraft epoch:
   Line_time = TC_Day*86400 + TC_MSec/1000 + TC_Micro/1e6
   Note that an IEEE 754 double precision (64-bit) number with a 52-bit fraction should provide sufficient precision to represent time differences from 01JAN2000 to 01JAN2050 with microsecond accuracy (1.6e15 microseconds < 2^51).
2. Validate and correct the image time codes as follows:
   a. Loop through the time codes from 1 to N-1, where N is the number of image data frames/time codes, and test the difference between the current and previous time codes against the nominal frame time from the CPF using the #define tolerance DTIME_TOL. The first of two consecutive time codes that are within the tolerance is the first valid time code. The DTIME_TOL value should be a CPF parameter.
   b. Initialize the TIRS clock model by setting the least squares variables to zero: $A_{00} = A_{01} = A_{11} = L_0 = L_1 = 0$
      i. Since the normal equation matrix, A, is symmetric, $A_{10} = A_{01}$ so it is not computed separately.
      ii. Add the first valid time code observation by adding 1 to A_{00}. This is all that is required since, by definition, the index difference and time difference (see below) are zero at the first valid point.
   c. For each subsequent time code:
      i. Compare the time code difference to a larger outlier tolerance (OUTLIER_TOL) chosen to bound the possible drift in the TIRS clock relative to the spacecraft clock (currently set to 50 microsec). The OUTLIER_TOL value should be a CPF parameter.
      ii. If the time code difference is within the outlier range, add the current time to a least squares linear TIRS clock model:
         1. Δnum = current index – first valid index
         2. Δtime = current time – first valid time
         3. Accumulate:
            a. Valid point count: A_{00} += 1
            b. Index difference: A_{01} += Δnum
            c. Squared index diff: A_{11} += Δnum*Δnum
            d. Time difference: L_0 += Δtime
            e. Time diff*index diff: L_1 += Δnum*Δtime
   d. Once all time codes have been analyzed, solve for the linear TIRS clock model parameters:
      i. determinant = $A_{00}*A_{11} – A_{01}*A_{01}$
      ii. If abs(determinant) <= 0.0 return an error
      iii. Offset = first valid time + $(A_{11}*L_0 – A_{01}*L_1)$ / determinant

iv. Rate = (A_{00}*L_1 – A_{01}*L_0) / determinant

e. Use the correction model to replace bad time codes:
   i. For each time code:
      1. Calculate the corresponding model time as:
         Mtime = Offset + (current index – first valid index) * Rate
      2. Calculate the actual time – model time difference.
         Diff = abs(time code – Mtime)
      3. Test the difference against DTIME_TOL
      4. If the difference exceeds DTIME_TOL, replace the current time code with the model value, Mtime
f. If no valid time codes were found, return an error.
g. Calculate the average observed frame time, delta_time, by subtracting the first valid/corrected time code from the last valid/corrected time code and dividing by the number of time codes minus one.
h. Store delta_time in the model.

3. Compute the image start time: image_start = line_time[0]
4. Ensure that the ancillary data ephemeris covers the image:
   a. Convert the ephemeris epoch to TAI seconds from spacecraft epoch:
      i. Load the leap second table from the CPF.
      ii. Search the leap second table for the last entry that is not later than the ephemeris epoch.
      iii. Add the total leap second count for that entry to the UTC date/time to yield TAI date/time.
      iv. Subtract the spacecraft TAI epoch to compute ephem_start in TAI seconds since the spacecraft epoch.
   b. Check the beginning of the ephemeris interval against the image start time (which is also TAI seconds since the spacecraft epoch):
      If (image_start – ephem_start) < 4 seconds
      Then report error “Ephemeris data does not cover the image” and exit
   c. Check the end of the ephemeris interval against the image stop time:
      ephem_stop = ephem_start + ephem_time[M-1] where M is the number of ephemeris data entries.
      image_stop = image_start + delta_time*(N-1)
      If (ephem_stop – image_stop) < 4 seconds
      Then report error “Ephemeris data does not cover the image” and exit
   d. Note that the 4 second overlap threshold would be a good thing to put in a #define statement as suggested below.
5. Repeat step 4 using the ancillary attitude data in place of the ephemeris data.
6. Compute the image start UTC epoch by converting image_start to UTC as described under “Convert Spacecraft Time Code to UTC” in the Ancillary Data Preprocessing Algorithm (4.1.4). This epoch will be stored as: Year, Day of Year, Seconds of Day.
7. Make sure the epoch is consistent with the ancillary data:
   a. If image_year > ephem_year or image_day > ephem_day
      Then image_year = ephem_year
           image_day = ephem_day

image_seconds = image_seconds + 86400

This ensures that all computations for a given imaging interval are based on the same day and, hence, on the same UT1UTC, pole wander, and leap second corrections.

8. Subtract the image start time from the line time codes so that the times are seconds from image start.
9. Store the image start UTC epoch (image_year, image_day, image_seconds) and the image offset times in the model structure.
10. Report/trend the results of the time code processing including:
    a. WRS Path/Row (input parameters)
    b. Image UTC epoch (year, day, seconds of day)
    c. L0R ID (input parameter)
    d. Work order ID (input parameter)
    e. Computed frame time (delta_time)
    f. Number of replaced time codes (bad_image_time_count)

Extract the integration time field from the Level 0R/1R image line header data. There will be one value for each image line. Convert the integration time from the first valid TIRS line header record to units of seconds and store in the model structure.

Extract and Process SSM Data

To populate the SSM model portion of the TIRS geometric model it is necessary to construct two elements: the SSM alignment matrix **M'**($\Delta r, \Delta p+2\Delta\gamma, \Delta y$) which is a function of the SSM alignment parameters from the CPF; and a sequence of SSM pointing angles and associated times. The SSM pointing angles are computed using the SSM telemetry data contained in the Level 0R ancillary data. The TIRS ancillary data group is provided every second. The relevant contents of the TIRS ancillary data group are shown in Table 4-35. Each 1 Hz group contains a time code and twenty-one (L8) or twenty-two (L9) 24-bit resolution SSM encoder samples.

Field	Size	Contents
Day	16 bits	Days since spacecraft epoch
Milliseconds	32 bits	Milliseconds of day
Microseconds	16 bits	Microseconds of millisecond
SSM Position 1-21	24 bits	24-bit resolution SSM encoder readout

Table 4-35. TIRS Scene Select Mirror Ancillary Data

The extra twenty-first and twenty-second samples are included, even though the encoder is sampled at 20 Hz, to ensure that encoder samples do not accumulate in the ancillary data output buffer. This means that the twenty-first and twenty-second samples will likely contain zeros in both the high order and low order words. For L8, due to variations in the encoder data generation and ancillary buffer output timing, the high- and low-order encoder data words can become misaligned, and one or both of the data words in sample #21 will be non-zero. This condition must be detected and corrected by the SSM data processing logic. For L9, this is handled on-board and the first twenty encoder samples should always be valid and ready for use.

Retrieve the SSM alignment angle and mirror angle deviation parameters from the CPF and construct the SSM alignment matrix, **M'**, using the equations above. Store the resulting alignment matrix in the SSM model. Determine which encoder (A or B) is active by performing a majority vote on the MCE bits (bits 0 and 1) of the elec_enabled_flags status word (for L8) or the T_MC_ENC_SEL telemetry field (for L9). If the number of status words with bit 1 set exceeds the number with bit 0 set, make the SSM reference angle equal to the CPF side B mirror nadir angle. Otherwise, use the side A value.

For L9, check the SSM operating mode flag (T_MC_MD_ME_SM) to see if it is the nominal absolute pointing control mode (4) or the contingency relative pointing control mode (10). The contingency mode 10 would only be used if loss-of-index anomalies become a problem during on-orbit operations. If the mirror is operating in mode 10, update the 12 high order bits of the encoder position telemetry samples using the commanded nadir position value for the active encoder, N_A or N_B, as follows:

1. Separate the commanded position (from the CPF) into the high 12 and low 12 bits. All values are signed 32 bit integers.
   $N_{high} = N_{A/B}$ & 0x00FFF000
   $N_{low} = N_{A/B}$ & 0x00000FFF
2. Separate out the 12 low order bits from the encoder position value:
   D_{low} = Encoder Position Value & 0x00000FFF
3. Subtract the commanded low order bits from the measured low order bits:
   Delta = $D_{low} - N_{low}$
4. Adjust the difference to put it in the range -2047 to +2048 counts:
   If Delta > 2048 then Delta -= 4096
   If Delta < -2047 then Delta += 4096
5. Add the difference to the commanded position:
   Corrected Encoder Position Value = $N_{A/B}$ + Delta

Quality check the entire TIRS SSM telemetry set as follows:

1. Validate the SSM telemetry time codes:
   a. Find the first valid time code as the first time code for which the time difference between it and the following time code is the nominal 1.0 second sampling interval, to within a pre-defined tolerance.
   b. Use the valid time code to correct all previous time codes using the nominal sampling interval.
   c. Use the valid time code to check all subsequent time codes using the nominal sampling interval. Any time codes failing the sampling interval tolerance are corrected using the previous valid sample time and the nominal sampling interval.
2. Due to the asynchronous SSM telemetry generation and ancillary data assembly processes, the L8 SSM encoder samples will sometimes be improperly aligned with the 1 Hz ancillary data frames. The encoder telemetry generation logic can run either faster or slower than nominal, leading to variations in the number of

samples accumulated in the output buffer between 1 second buffer read operations. Due to these variations in the encoder sample timing, any given ancillary data frame may contain from 19 to 21 encoder samples. The TIRS ancillary data assembly logic was modified to read (and output) 21 (L8) or 22 (L9) samples for each frame to ensure that any extra encoder samples do not accumulate in the encoder telemetry buffer. Furthermore, for L8 data the upper 16 bits and lower 16 bits of each encoder value are buffered separately, so a given frame may have partial encoder samples (only upper 16 or only lower 16). The following logic, necessary for L8 only, is designed to align the upper and lower data words of the SSM telemetry encoder samples in each ancillary data record:

a. If encoder sample 20 is zero and encoder sample 21 is zero
    i. If this is the last ancillary record, set sample 20 equal to sample 19
    ii. Otherwise set sample 20 equal to sample 1 from the next record.
b. If the high 16-bits of sample 20 are zero:
    i. If the high 16-bits of sample 21 are non-zero, move the bits from sample 21 to sample 20.
    ii. Otherwise, move the high 16-bits from the first sample in the next record to sample 20, and move all the high order words in the next record up one sample (setting the 21st sample to zero). If the current record is the last record, copy sample 19 into sample 20.
c. If the low 16-bits of sample 20 are zero:
    i. If the low 16-bits of sample 21 are non-zero, move the bits from sample 21 to sample 20.
    ii. Otherwise, move the low 16-bits from the first sample in the next record to sample 20, and move all the low order words in the next record up one sample. If the current record is the last record, copy sample 19 into sample 20.
d. If sample 21 is equal to zero, go to the next record.
e. If only the high 16-bits of sample 21 are zero, and this is not the last record, move all the high order words in the next record up one sample.
f. If only the low 16-bits of sample 21 are zero, and this is not the last record, move all the low order words in the next record up one sample.

3. Extract the first 20 samples in each record for subsequent processing. Note that this method discards extra samples and fills missing samples by duplicating either the next or the previous sample. For L9, the first 20 samples should always be valid and no check for missing samples or high/low word slipping is necessary. In this case, the extra samples are included solely for diagnostic purposes.
4. Validate the 24-bit SSM telemetry encoder readings:
    a. Find the first valid encoder reading as the first reading for which the difference between it and the SSM reference angle (nominal nadir pointing angle for the current MCE side for L8 or difference from the commanded nadir position for L9, determined above) is less than the quality tolerance specified in the CPF. Note that the quality tolerance is nominally 20 micro-

radians which is much smaller than the potential offset between the commanded nadir position and the calibrated nadir position.
   b. Use the valid angle to replace all previous 24-bit encoder readings.
   c. Use the valid angle to check and, if necessary, replace all subsequent 24-bit encoder readings, by comparing each value to the previous, valid/corrected, value.
5. Although an anomaly with the SSM encoder that occasionally rendered the upper 14 bits of the read out invalid, was observed during TIRS-1 subsystem level test, this behavior has not been seen in the integrated TIRS-1 instrument. Due to the low probability of this being a problem, no special logic was added to handle this case for L8. Instead, standard outlier detection and correction logic was relied upon to correct this problem if it occurs. For the L9 TIRS-2, the mode 10 contingency was added so additional logic to correct the high order bits, described above, has been included. The outlier detection step should still correct any cases where the upper 12-bit replacement logic fails.

The quality-checked TIRS ancillary data groups are examined to find the range of samples that correspond to the TIRS image.

1. Scan through the TIRS ancillary data, and convert each time code to seconds from spacecraft epoch.
2. Find the last ancillary data record with a time code that is earlier than the TIRS image start time. This is the first TIRS ancillary data packet to extract.
3. Find the first ancillary data record with a time code that is later than the TIRS image stop time. This is the last TIRS ancillary data packet to extract.
4. Extract the SSM (mirror control electronics or MCE) telemetry fields and time codes for the ancillary data records covering the TIRS image.

For each extracted SSM telemetry group:

1. Convert the TIRS ancillary data time code to seconds from spacecraft epoch and find the difference between the ancillary data time and the TIRS image start time.
2. Add the SSM encoder time offset (from the CPF) to the sample time so that it represents the time of the first SSM encoder sample.
3. Compute the sample times for the remaining samples by adding increments of 0.05 seconds to the previous sample time.
4. Load the SSM encoder samples into signed 32-bit integer variables for subsequent manipulation.
5. Scale the encoder counts to radians and subtract the SSM reference angle (set as described above):
   a. Angle = 2*pi*Sample_Value/0x01000000 – θ_0
   b. If Angle > pi Then Angle -= 2*pi
   c. If Angle < -pi Then Angle += 2*pi

   Note: These angles should all be close to zero for Earth viewing.
6. Add the 20 computed times (from image start) and the 20 SSM angles to the SSM model.

Smooth the SSM angles as follows:

1. For each SSM angle:
    a. Find the previous two and next two original data points, or the closest four points if two cannot be found before and after.
    b. Compute the average of the current point and the four closest points.
    c. The smoothed value is the mean of the current point value and the 4 closest points.
2. Store the smoothed sequence of SSM angles in the SSM model.

The baseline TIRS geometric algorithms assume no significant temperature dependence in either the SSM or in TIRS alignment. We do assume that the temperature telemetry present in the TIRS ancillary data will be recorded in the trending database (for radiometric purposes) so that pointing temperature sensitivities could be studied on-orbit (see note #8).

Extract Ancillary Ephemeris and Attitude Data
The subset of ancillary ephemeris and attitude data needed to span the image data are extracted from the input preprocessed ancillary data stream and stored in the model structure. Extra ancillary data, nominally 4 seconds, is required before and after the image start/stop times to ensure model continuity from scene to scene within an imaging interval. This ancillary data overlap time parameter could be stored in a #define statement as it would not be expected to change once established.

The ephemeris data extraction/subsetting procedure is as follows:

1. Compute the time offset from the ephemeris epoch time to the desired ephemeris start time for this image. Note that since the image epoch has been adjusted to fall in the same day as the ephemeris epoch this can be done using the seconds of day fields only.
    ephem_start = image_seconds – ancillary_overlap – ephem_seconds
    Noting that image_seconds and ephem_seconds are the seconds of day fields from the image and ephemeris epoch times, respectively.
2. Loop through the ephemeris sample times to find the last entry that does not exceed ephem_start. This is the ephemeris start index (eph_start_index).
3. Compute the time offset from the ephemeris epoch time to the desired ephemeris stop time for this image.
    ephem_stop = image_seconds + line_time[N-1] + ancillary_overlap – ephem_seconds
    N is the number of image lines, and N-1 is the index of the last image line time.
4. Loop through the ephemeris sample times to find the first entry that exceeds ephem_stop. This is the ephemeris stop index (eph_stop_index).
5. Compute a new ephemeris UTC epoch for this image:
    imgeph_year = ephem_year
    imgeph_day = ephem_day

imgeph_seconds = ephem_seconds +
ephem_samp_time[eph_start_index]

6. Load the ECI and ECEF ephemeris samples from eph_start_index to eph_stop_index (inclusive) into the model structure, adjusting the sample times so that they are offset from the UTC epoch computed in step 5.

The attitude data extraction/subsetting procedure is as follows:

1. Compute the time offset from the attitude epoch time to the desired attitude start time for this image. Note that since the image epoch has been adjusted to fall in the same day as the ancillary data (ephemeris and attitude) epochs this can be done using the seconds of day fields only.
   att_start = image_seconds – ancillary_overlap – att_seconds
   Noting that image_seconds and att_seconds are the seconds of day fields from the image and attitude epoch times, respectively.
2. Loop through the attitude sample times to find the last entry that does not exceed att_start. This is the attitude start index (att_start_index).
3. Compute the time offset from the attitude epoch time to the desired attitude stop time for this image.
   att_stop = image_seconds + line_time[N-1] + ancillary_overlap – att_seconds
4. Loop through the attitude sample times to find the first entry that exceeds att_stop. This is the attitude stop index (att_stop_index).
5. Compute a new attitude UTC epoch for this image:
   imgatt_year = att_year
   imgatt_day = att_day
   imgatt_seconds = att_seconds + att_samp_time[att_start_index]
6. For Earth-view acquisitions, load the roll-pitch-yaw samples from att_start_index to att_stop_index (inclusive) into the model structure, adjusting the sample times so that they are offset from the UTC epoch computed in step 5.
7. For lunar/stellar acquisitions, convert the ECI quaternion samples from att_start_index to att_stop_index (inclusive) to ECI roll-pitch-yaw values, as described below, and store the computed roll-pitch-yaw values in the model structure, adjusting the sample times so that they are offset from the UTC epoch computed in step 5.

Converting ECI Quaternions to Roll-Pitch-Yaw

For lunar and stellar acquisitions, the ECI attitude representation is stored in the model structure. In the baseline model, this is done by converting the ECI quaternions to roll-pitch-yaw values relative to the ECI axes. This is one of the motivations for considering a transition to using a quaternion attitude representation in the model in the future.

The ECI quaternions are converted to roll-pitch-yaw values as follows:

1. Compute the rotation matrix corresponding to the ECI quaternion values:

$\mathbf{M}_{ACS2ECI}$ =

$$\begin{bmatrix} q_1^2 - q_2^2 - q_3^2 + q_4^2 & 2(q_1q_2 - q_3q_4) & 2(q_1q_3 + q_2q_4) \\ 2(q_1q_2 + q_3q_4) & -q_1^2 + q_2^2 - q_3^2 + q_4^2 & 2(q_2q_3 - q_1q_4) \\ 2(q_1q_3 - q_2q_4) & 2(q_2q_3 + q_1q_4) & -q_1^2 - q_2^2 + q_3^2 + q_4^2 \end{bmatrix}$$

2. Compute the corresponding ACS to ECI roll-pitch-yaw values:

$$\text{roll}' = -\tan^{-1}\left(\frac{M_{2,1}}{M_{2,2}}\right)$$

$$\text{pitch}' = \sin^{-1}\left(M_{2,0}\right)$$

$$\text{yaw}' = -\tan^{-1}\left(\frac{M_{1,0}}{M_{0,0}}\right)$$

   Note that in implementing these calculations it is important to use the ATAN2 rather than the ATAN arctangent implementation in order to retain the correct quadrants for the Euler angles. This is not a concern in Earth-view imagery where the angles are always small, but becomes an issue for these lunar/stellar ACS to ECI angles.
3. Store the ECI roll-pitch-yaw values in the model attitude data table.

At the completion of this sub-algorithm the model structure contains the image frame time stamps, the multispectral and panchromatic sample and integration times, the ancillary ephemeris data, in both ECI and ECEF representations, covering the image, and the ancillary attitude data covering the image.

Jitter Correction Data Preprocessing

Jitter correction preprocessing operates on the roll-pitch-yaw attitude data stream extracted from the spacecraft ancillary data to separate the low frequency spacecraft pointing effects from the higher frequency jitter disturbances. The low frequency pointing model is used for line-of-sight projection and other geolocation processing while the high frequency jitter effects are applied as per-line corrections during image resampling. To implement this frequency separation in the line-of-sight model the original attitude sequence is passed through a low pass filter with a cutoff frequency defined as a parameter in the CPF. This cutoff frequency will nominally be in the 1 Hz to 10 Hz range. The value ultimately selected for this cutoff frequency will depend upon the actual disturbance profile observed in the spacecraft attitude data. The high frequency data stream should be limited in magnitude to subpixel (ideally sub-half-pixel) effects, but the lower the cutoff frequency can be, the sparser (and smaller) the TIRS resampling grid can be made in the line (time) dimension.

The low pass filtered version of the attitude sequence is differenced with the original data to construct the complementary high pass data sequence. The high pass sequence is then interpolated at the TIRS image line times to provide a table containing high frequency roll-pitch-yaw corrections for each image line. This jitter table is stored in the TIRS line-of-sight model. The original attitude sequence in the line-of-sight model is

replaced with the low pass filtered sequence to avoid double counting the high frequency effects. This process is depicted in Figure 4-61.

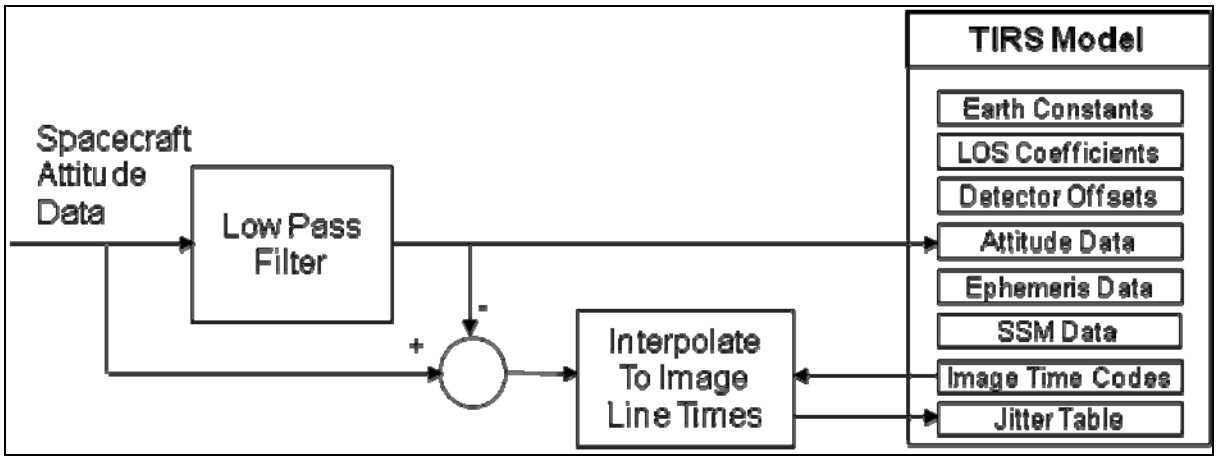

Figure 4-61. Jitter Correction Table Generation Data Flow

The jitter table construction processing sequence is as follows:

1. Extract a copy of the original attitude data sequence from the TIRS line-of-sight model.
2. Retrieve the low pass filter cutoff frequency from the CPF.
3. Design a low pass filter with the desired cutoff frequency and apply it to the attitude data.
    a. Use the cutoff frequency and attitude data sampling time to compute the size of the desired filter as follows:
        i. Compute the normalized cutoff frequency (the ratio of the cutoff frequency to the attitude data sampling frequency):
            n_cutoff = cutoff_frequency / attitude_sample_frequency
        Note that this is the same as:
            n_cutoff = cutoff_frequency * attitude_sample_time
        ii. Compute the number of samples per cycle at the cutoff frequency:
            Nsamp = 1 / n_cutoff
        iii. Multiply the number of samples per cycle by 3 and add 1 to yield the desired filter size:
            FSize = 3*Nsamp + 1
        iv. If this results in an even filter size, add one:
            If (FSize modulo 2 == 0) FSize = FSize + 1
    b. Use the Remez exchange algorithm to design the filter and generate the filter weights. The standard Parks-McClellan finite impulse response (FIR) digital filter design method uses the Remez exchange algorithm (ref. Theory and Application of Digital Signal Processing, Rabiner and Gold, Prentice-Hall, 1975). A C implementation of this algorithm called remez.c, authored by Jake Janovetz at the University of Illinois, is available under

the GNU Public License. This implementation specifies the desired (low pass, in this case) filter response using the following parameters:

i. Filter size (number of taps) – FSize computed in item a. above.
ii. Number of frequency bands to use – 2, one pass band (low frequency) and one stop band (high frequency).
iii. Band frequency bounds – 0 to the normalized cutoff frequency (n_cutoff) for the pass band and 1.5*n_cutoff to 0.5 (normalized Nyquist frequency) for the stop band.
iv. Desired band gains – 1 for pass band (low) and 0 for stop band (high).
v. Band weights (how tightly to constrain the actual filter response to the design filter response in each band) – 1 for pass band and 10 for stop band.
vi. Filter type – BANDPASS (the low pass filter is a special case of the more general BANDPASS filter type supported by the remez algorithm.

c. Make sure the synthesized filter is normalized (weights sum to 1) by adding the filter tap values and dividing each tap by the total.

```
sum = Σ h[i]   where h[i] are the FSize filter taps.
h'[i] = h[i] / sum for i = 1 to FSize.
```

d. Convolve the filter with the roll-pitch-yaw attitude data one axis at a time:

```
half_size = FSize / 2
for index = 0 to num_rpy – 1
    low_roll[index] = low_pitch[index] = low_yaw[index] = 0
    for ii = -half_size to half_size
        if ( index + ii < 0 ) j = -index – ii
        else if ( index + ii < num_rpy ) j = index + ii
        else j = 2*num_rpy – index - ii – 1
    low_roll[index] += roll[j]*h[ii+half_size]
    low_pitch[index] += pitch[j]*h[ii+half_size]
    low_yaw[index] += yaw[j]*h[ii+half_size]
```

4. Subtract the low pass filtered sequences from the original sequences to extract the high frequency portion of the data, and transfer any residual bias (non-zero mean value) from the imaging portion of the high frequency sequence to the low frequency sequence:

```
roll_bias = pitch_bias = yaw_bias = 0
att_pts = 0
for index = 0 to nrpy–1
    high_roll[index] = roll[index] – low_roll[index]
    high_pitch[index] = pitch[index] – low_pitch[index]
    high_yaw[index] = yaw[index] – low_yaw[index]
    if ( image_start_time < attitude_time[index] < image_stop_time )
        roll_bias += high_roll[index]
        pitch_bias += high_pitch[index]
        yaw_bias += high_yaw[index]
        att_pts += 1
```

roll_bias = roll_bias / att_pts
pitch_bias = pitch_bias / att_pts
yaw_bias = yaw_bias / att_pts
for index = 0 to nrpy-1
high_roll[index] -= roll_bias
low_roll[index] += roll_bias
high_pitch[index] -= pitch_bias
low_pitch[index] += pitch_bias
high_yaw[index] -= yaw_bias
low_yaw[index] += yaw_bias

5. Interpolate the high frequency sequence values at the TIRS line sampling times to create the model jitter table:
   For each TIRS image line = 0 to number of lines:
   Compute the line sampling time as:
   index = line
   line_time = line_time_stamp[index]
   + integration_time/2
   Convert to time from attitude epoch:
   line_time += image_epoch – attitude _epoch
   Interpolate high frequency roll-pitch-yaw values at this time using four point Lagrange interpolation:
   Compute starting index for interpolation:
   index = floor(line_time / attitude_sample_time) – 1
   Compute the fractional sample offset to the line time:
   w = line_time / attitude_sample_time – index – 1
   Compute the Lagrange weights:
   w1 = -w * (w – 1) * (w – 2) / 6
   w2 = (w + 1) * (w – 1) * (w – 2) / 2
   w3 = -w * (w + 1) * (w – 2) / 2
   w4 = (w + 1) * w * (w – 1) / 6
   Interpolate:
   roll = high_roll[index]*w1 + high_roll[index+1]*w2
   + high_roll[index+2]*w3 + high_roll[index+3]*w4
   pitch = high_pitch[index]*w1 + high_pitch[index+1]*w2
   + high_pitch[index+2]*w3 + high_pitch[index+3]*w4
   yaw = high_yaw[index]*w1 + high_yaw[index+1]*w2
   + high_yaw[index+2]*w3 + high_yaw[index+3]*w4
6. Replace the original model attitude data sequence with the low pass filtered attitude data sequence.

Note that if TIRS and OLI processing is combined, the attitude filtering and high-pass/low-pass separation logic should be common, but the two sensors would still require their own jitter tables since these tables are based on the image line times, which are different for TIRS and OLI.

Process LOS Model Sub-Algorithm
This function loads the LOS Legendre polynomial coefficients and other model components from the CPF, and performs additional processing on the attitude and ephemeris information in the LOS model structure. It invokes the following sub-algorithms.

Read CPF Model Parameters Sub-Algorithm
This function loads model components from the CPF. In the heritage ALIAS implementation some of these model components either did not exist (e.g., instrument offset from spacecraft center of mass) or were used for image resampling but not LOS model computations (e.g., detector offset table) and so, were not included in the model. These are included in the TIRS model to make it self-contained for purposes of line-of-sight computations.

CPF parameters loaded into the geometric model include the following:

1. Earth orientation parameters – the UT1UTC and pole wander (x,y) parameters for the current day are stored in the model to avoid the necessity of repeatedly looking them up in the CPF.
2. TIRS offset from spacecraft center of mass – a 3-vector that captures the small offset, in spacecraft body coordinates, between the TIRS instrument, where images are captured, and the spacecraft center of mass, the position of which is reported in the ancillary ephemeris data, making it possible to translate the ephemeris data to the TIRS. Technically, this would be the vector from the spacecraft center of mass to the center of the TIRS entrance pupil. Note that this formulation assumes that the spacecraft on-board GPS data processing includes the GPS to spacecraft center of mass (CM) offset and that the spacecraft is, in fact, reporting CM positions not GPS antenna positions. If the ephemeris represents the GPS antenna location then we would need to know the spacecraft CM to GPS antenna offset as well.
3. TIRS to ACS alignment matrix – a 3-by-3 matrix that captures the relative orientation of the TIRS coordinate system to the ACS coordinate system, making it possible to rotate the TIRS instrument-space line-of-sight vectors into the ACS reference system. In the heritage ALIAS system this was actually represented in the CPF by an ACS to instrument rotation matrix, which was inverted for each LOS model invocation. Whichever convention is used in the CPF, the LOS model should store the TIRS-to-ACS rotation matrix.
4. TIRS line-of-sight Legendre polynomials – a set of 8 coefficients (4 along-track and 4 across-track) for each band on each SCA. Each set of 4 forms a 3rd order Legendre polynomial that is used to evaluate a nominal LOS angle (along- or across-track) for the detectors in that band on that SCA. This differs from the OLI implementation, which used a 2nd order model (see the Read LOS Vectors Sub-Algorithm description below).
5. TIRS detector delay table – a table consisting of two values (along- and across-track) per detector reflecting the offset of each actual detector from its nominal location (as modeled by the 3rd order Legendre polynomials – see below). In the

heritage ALIAS implementation these were small subpixel offsets that were applied in the image resampling procedure. With the TIRS, this table will also contain any offsets due to detector deselect/replacement (i.e., the operational use of a detector from one of the redundant rows). This table is needed in those LOS projection algorithms that utilize either actual (whole pixel offsets) or exact (full subpixel offsets) detector locations.

Read LOS Vectors Sub-Algorithm
This function retrieves the line of sight vectors from the CPF. The line of sight vectors are stored as sets of 3rd order Legendre polynomial coefficients. There is a unique set of 8 coefficients for each band of each SCA, 4 for the along-track polynomial and 4 for the across-track polynomial. These values are read from the CPF and stored in the LOS model. The polynomials are used to compute along- and across-track viewing angles for each nominal detector.

Initialize the Precision Model Sub-Algorithm
This function initializes the precision LOS correction model parameters. If the optional precision model input parameters are provided, those values are used. In the normal case, those parameters are absent and the correction model is initialized as follows:

Set the precision correction reference time to the center of the scene:
t_ref = line_time[N/2] where: N is the number of time codes in the image
Set the ephemeris correction model order to zero: eph_order = 0
Set both ephemeris X correction parameters to zero:
x_corr[0] = 0.0, x_corr[1] = 0.0
Set both ephemeris Y correction parameters to zero:
y_corr[0] = 0.0, y_corr[1] = 0.0
Set both ephemeris Z correction parameters to zero:
z_corr[0] = 0.0, z_corr[1] = 0.0
Set the attitude correction model order to zero: att_order = 0
Set all three attitude roll correction parameters to zero:
roll_corr[0] = 0.0, roll_corr[1] = 0.0, roll_corr[2] = 0.0
Set all three attitude pitch correction parameters to zero:
pitch_corr[0] = 0.0, pitch_corr[1] = 0.0, pitch_corr[2] = 0.0
Set all three attitude yaw correction parameters to zero:
yaw_corr[0] = 0.0, yaw_corr[1] = 0.0, yaw_corr[2] = 0.0

Note that these parameters are used to compute the corrected ephemeris and attitude data sequences, which are also stored in the model. The parameters themselves are included in the model primarily to document the magnitude of the corrections applied and to facilitate more advanced uses of the model creation logic. For example, it is sometimes useful to be able to force a particular model bias (e.g., a roll angle) into a model that is to be used for data simulation (see note 6). Therefore, though not strictly necessary for operational data processing, these parameters aid in anomaly resolution, data simulation, and algorithm development. In normal operations, these initial correction parameters are all zero and the "corrected" attitude and ephemeris data

sequences are identical to the "original" attitude and ephemeris data prior to the execution of the LOS model correction algorithm. Subsequent algorithms (e.g., LOS projection) operate on the corrected data.

Correct Attitude Sub-Algorithm

This function applies the ACS/body space attitude corrections computed by the LOS/precision correction procedure to the attitude data sequence. It outputs a parallel table of roll-pitch-yaw values with the precision corrections applied. In the model creation context the precision corrections are zero so the two sets of attitude data are identical. Though applying the precision corrections to construct the corrected attitude sequence could be said to be overkill for model creation (since the corrections are nominally zero at this point) this capability is required for LOS model correction and is used here to support the use of the model creation algorithm for data simulation and anomaly resolution as it makes it possible to force initial biases into the model. This sub-algorithm will also be used by the LOS/precision correction algorithm to create the precision model. Note that the formulation is somewhat different for Earth-view scenes (Acquisition Type = Earth) than it is for lunar and stellar observations.

Earth Scenes

For Earth-view scenes the sequence of transformations required to convert a line-of-sight in the TIRS instrument coordinate system, generated using the Legendre polynomials, is:

$$\underline{\mathbf{x}}_{ECEF} = \mathbf{M}_{ORB2ECEF}\,\mathbf{M}_{ACS2ORB}\,\mathbf{M}_{Precision}\,\mathbf{M}_{TIRS2ACS}\,\mathbf{M}_{SSM}(\theta)\,\mathbf{M}_{Tele2SSM}\,\underline{\mathbf{x}}_{TIRS}$$

where:

$\underline{\mathbf{x}}_{TIRS}$ is the Legendre-derived instrument LOS vector
$\mathbf{M}_{Tele2SSM}$ is the TIRS telescope alignment matrix described above
$\mathbf{M}_{SSM}(\theta)$ is the SSM reflection matrix, described above, which is a function of SSM angle
$\mathbf{M}_{TIRS2ACS}$ is the TIRS to ACS alignment matrix from the CPF
$\mathbf{M}_{Precision}$ is the correction to the attitude data computed by the LOS/precision correction procedure
$\mathbf{M}_{ACS2ORB}$ is the spacecraft attitude (roll-pitch-yaw)
$\mathbf{M}_{ORB2ECEF}$ is the orbital to ECEF transformation computed using the ECEF ephemeris
$\underline{\mathbf{x}}_{ECEF}$ is the LOS vector in ECEF coordinates

Since TIRS will occasionally be viewing off-nadir and it is more natural to model attitude errors in the ACS/body coordinate system than in the orbital coordinate system, the order of the $\mathbf{M}_{ACS2ORB}$ and $\mathbf{M}_{Precision}$ rotations have been reversed for L8/9 as compared to the heritage Landsat/EO-1 implementation. The impact is minimal in the model and LOS projection but becomes more important for the LOS/precision correction algorithm.

This new sub-algorithm pre-computes the $\mathbf{M}_{ACS2ORB}$ $\mathbf{M}_{Precision}$ combination and stores the corresponding corrected roll-pitch-yaw attitude sequence in the model structure. This approach has several advantages:

1. It streamlines the application of the model for LOS projection by removing the step of explicitly applying the precision correction.
2. It allows for the use of a more complex correction model in the future since the application of the model is limited to this unit. Note that the Earth-view attitude correction model consists of the following model parameters:
   Precision reference time: t_ref in seconds from the image epoch (at the center of the image time window)
   Attitude model order: att_order = 1
   Roll bias and rate corrections: roll_corr[] = roll_bias, roll_rate
   Pitch bias and rate corrections: pitch_corr[] = pitch_bias, pitch_rate
   Yaw bias and rate corrections: yaw_corr[] = yaw_bias, yaw_rate
   This model is dealt with in more detail in the line-of-sight correction algorithm description.
3. Retaining both the original and corrected attitude sequences in the model make the model self-contained and will make it unnecessary for the LOS/precision correction algorithm to access the preprocessed ancillary data.

The disadvantage is that it doubles the size of the attitude data in the model structure.

The construction of the corrected attitude sequence proceeds as follows:

For each point in the attitude sequence j = 0 to K-1:

1. Compute the rotation matrix corresponding to the j^{th} roll-pitch-yaw values:

$\mathbf{M}_{ACS2ORB}$ =

$$\begin{bmatrix} \cos(p)\cos(y) & \sin(r)\sin(p)\cos(y)+\cos(r)\sin(y) & \sin(r)\sin(y)-\cos(r)\sin(p)\cos(y) \\ -\cos(p)\sin(y) & \cos(r)\cos(y)-\sin(r)\sin(p)\sin(y) & \cos(r)\sin(p)\sin(y)+\sin(r)\cos(y) \\ \sin(p) & -\sin(r)\cos(p) & \cos(r)\cos(p) \end{bmatrix}$$

2. Compute the precision correction at the time (t_att = att_seconds + att_time[j]) corresponding to the attitude sample:
   a. $roll\_correction = \sum_{i=0}^{att\_order} \boldsymbol{roll\_corr}[i] * (t\_att - t\_ref - image\_seconds)^i$
   b. $pitch\_correction = \sum_{i=0}^{att\_order} \boldsymbol{pitch\_corr}[i] * (t\_att - t\_ref - image\_seconds)^i$
   c. $yaw\_correction = \sum_{i=0}^{att\_order} \boldsymbol{yaw\_corr}[i] * (t\_att - t\_ref - image\_seconds)^i$

   Note that only the seconds of day fields are needed for the attitude and image epochs as they are constrained to be based on the same year and day.
3. Compute the rotation matrix corresponding to roll_correction (r), pitch_correction (p), and yaw_correction (y) ($\mathbf{M}_{Precision}$) using the same equations presented in step 1 above.
4. Compute the composite rotation matrix: $\mathbf{M} = \mathbf{M}_{ACS2ORB}\ \mathbf{M}_{Precision}$
5. Compute the composite roll-pitch-yaw values:

$$\text{roll}' = -\tan^{-1}\left(\frac{M_{2,1}}{M_{2,2}}\right)$$

$$\text{pitch}' = \sin^{-1}\left(M_{2,0}\right)$$

$$\text{yaw}' = -\tan^{-1}\left(\frac{M_{1,0}}{M_{0,0}}\right)$$

6. Store the composite roll'-pitch'-yaw' values in the j^{th} row of the corrected attitude data table.

Lunar and Stellar Scenes

Though there is no TIRS requirement for lunar or stellar data processing, this capability is retained to maintain compatibility with the OLI geometric model. For celestial (lunar or stellar) observations the sequence of transformations required to convert a line-of-sight in the TIRS instrument coordinate system, generated using the Legendre polynomials, is:

$$\underline{\mathbf{x}}_{ECI} = \mathbf{M}_{ACS2ECI}\ \mathbf{M}_{Precision}\ \mathbf{M}_{TIRS2ACS}\ \mathbf{M}_{SSM}(\theta)\ \mathbf{M}_{Tele2SSM}\ \underline{\mathbf{x}}_{TIRS}$$

where:

$\underline{\mathbf{x}}_{TIRS}$ is the Legendre-derived instrument LOS vector
$\mathbf{M}_{Tele2SSM}$ is the TIRS telescope alignment matrix described above
$\mathbf{M}_{SSM}(\theta)$ is the SSM reflection matrix, described above, which is a function of SSM angle
$\mathbf{M}_{TIRS2ACS}$ is the TIRS to ACS alignment matrix from the CPF
$\mathbf{M}_{Precision}$ is the correction to the attitude data computed by the LOS/precision correction procedure
$\mathbf{M}_{ACS2ECI}$ is the spacecraft attitude in the ECI frame derived from the ECI quaternions in the preprocessed ancillary data
$\underline{\mathbf{x}}_{ECI}$ is the LOS vector in ECI coordinates

The advantage of modeling the precision attitude corrections in ACS rather than orbital coordinates becomes apparent here, since the orbital frame is not used in the lunar case.

This sub-algorithm pre-computes the $\mathbf{M}_{ACS2ECI}\ \mathbf{M}_{Precision}$ combination and stores the corresponding corrected attitude sequence (as roll-pitch-yaw values relative to ECI) in the model structure. Another difference between the Earth-view and lunar/stellar models is in the formulation of the precision model. The lunar attitude correction model adds an acceleration term to the Earth-view correction model parameters:

Precision reference time: t_ref in seconds from the image epoch (nominally near the center of the image time window)
Attitude correction model order: att_order = 2
Roll bias, rate, and acceleration corrections: roll_corr[] = roll_bias, roll_rate, roll_acceleration

Pitch bias, rate, and acceleration corrections: pitch_corr[] = pitch_bias, pitch_rate, pitch_acceleration
Yaw bias, rate, and acceleration corrections: yaw_corr[] = yaw_bias, yaw_rate, yaw_acceleration

Due to the different orders of the Earth-view and lunar correction models, this model is stored as an array in the model structure along with a field defining the model order. The precision model is dealt with in more detail in the line-of-sight correction algorithm description.

The processing steps to construct the corrected attitude sequence is the same for lunar/stellar acquisitions, although the interpretation of the roll-pitch-yaw values is slightly different, and proceeds as follows:

For each point in the attitude sequence j = 0 to K-1:

1. Compute the rotation matrix corresponding to the j[th] ECI roll-pitch-yaw values:

$\mathbf{M}_{ACS2ECI}$ =

$$\begin{bmatrix} \cos(p)\cos(y) & \sin(r)\sin(p)\cos(y)+\cos(r)\sin(y) & \sin(r)\sin(y)-\cos(r)\sin(p)\cos(y) \\ -\cos(p)\sin(y) & \cos(r)\cos(y)-\sin(r)\sin(p)\sin(y) & \cos(r)\sin(p)\sin(y)+\sin(r)\cos(y) \\ \sin(p) & -\sin(r)\cos(p) & \cos(r)\cos(p) \end{bmatrix}$$

2. Compute the precision correction at the time (t_att = att_seconds + att_time[j]) corresponding to the attitude sample:
   d. $roll\_correction = \sum_{i=0}^{att\_order} \boldsymbol{roll\_corr}[i] * (t\_att - t\_ref - image\_seconds)^i$
   e. $pitch\_correction = \sum_{i=0}^{att\_order} \boldsymbol{pitch\_corr}[i] * (t\_att - t\_ref - image\_seconds)^i$
   f. $yaw\_correction = \sum_{i=0}^{att\_order} \boldsymbol{yaw\_corr}[i] * (t\_att - t\_ref - image\_seconds)^i$

   Note that only the seconds of day fields are needed for the attitude and image epochs as they are constrained to be based on the same year and day.
3. Compute the rotation matrix corresponding to roll_correction (r), pitch_correction (p), and yaw_correction (y):

$\mathbf{M}_{Precision}$ =

$$\begin{bmatrix} \cos(p)\cos(y) & \sin(r)\sin(p)\cos(y)+\cos(r)\sin(y) & \sin(r)\sin(y)-\cos(r)\sin(p)\cos(y) \\ -\cos(p)\sin(y) & \cos(r)\cos(y)-\sin(r)\sin(p)\sin(y) & \cos(r)\sin(p)\sin(y)+\sin(r)\cos(y) \\ \sin(p) & -\sin(r)\cos(p) & \cos(r)\cos(p) \end{bmatrix}$$

4. Compute the composite rotation matrix: $\mathbf{M} = \mathbf{M}_{ACS2ECI}\ \mathbf{M}_{Precision}$
5. Compute the composite ACS to ECI roll-pitch-yaw values:

$$\text{roll}' = -\tan^{-1}\left(\frac{M_{2,1}}{M_{2,2}}\right)$$

$$\text{pitch}' = \sin^{-1}\left(M_{2,0}\right)$$

$$\text{yaw}' = -\tan^{-1}\left(\frac{M_{1,0}}{M_{0,0}}\right)$$

Note that in implementing these calculations it is important to use the ATAN2 rather than the ATAN arctangent implementation in order to retain the correct quadrants for the Euler angles. This is not a concern in Earth-view imagery where the angles are always small, but becomes an issue for these lunar/stellar ACS to ECI angles.

6. Store the composite roll'-pitch'-yaw' values in the j^{th} row of the corrected attitude data table.

Correct Ephemeris Sub-Algorithm

The heritage ALIAS function converts the ephemeris information (position and velocity) from the Earth-Centered Inertial (ECI J2000) system to the ECEF system and applies the ephemeris corrections computed in the LOS/precision correction procedure to both ephemeris sets. Since both ECI and ECEF representations of the ephemeris are now provided by the ancillary data preprocessing algorithm (4.1.4), the first portion of the heritage algorithm is no longer necessary (or could be reused in the ancillary data preprocessing algorithm). Though applying the precision corrections to construct the corrected ephemeris sequence could be said to be overkill for model creation (since the corrections are nominally zero at this point) this capability is required for LOS model correction and is used here to support the use of the model creation algorithm for data simulation and anomaly resolution as it makes it possible to force initial biases into the model. This sub-algorithm will also be used by the LOS/precision correction algorithm to create the precision model.

The precision correction parameters are stored in the LOS model in the spacecraft orbital coordinate system as three position (x_bias, y_bias, z_bias) corrections and three velocity (x_rate, y_rate, z_rate) corrections that, like the attitude corrections, are relative to t_ref. These values must be converted to the ECEF and ECI coordinate systems. Once the precision correction is determined in the ECEF/ECI coordinate system, the ECEF/ECI ephemeris values can be updated with the precision parameters.

Loop on LOS model ephemeris points j = 0 to N-1

Compute the precision correction:

Calculate delta time for precision correction:

dtime = ephem_seconds + ephem_time[j] – t_ref – image_seconds

Calculate the change in X, Y, Z due to precision correction. Corrections are in terms of spacecraft orbital coordinates.

dx orb = model precision x_corr[0] + model precision x_corr[1] * dtime
dy orb = model precision y_corr[0] + model precision y_corr[1] * dtime
dz orb = model precision z_corr[0] + model precision z_corr[1] * dtime

where:
model precision x_corr[0] = precision (orbital) update to X position
model precision y_corr[0] = precision (orbital) update to Y position
model precision z_corr[0] = precision (orbital) update to Z position
model precision x_corr[1] = precision (orbital) update to X velocity
model precision y_corr[1] = precision (orbital) update to Y velocity
model precision z_corr[1] = precision (orbital) update to Z velocity

Construct precision position and velocity “delta” vectors.

$$[\text{dorb}] = \begin{bmatrix} \text{dx orb} \\ \text{dy orb} \\ \text{dz orb} \end{bmatrix}$$

$$[\text{dvorb}] = \begin{bmatrix} \text{model precision x\_corr}[1] \\ \text{model precision y\_corr}[1] \\ \text{model precision z\_corr}[1] \end{bmatrix}$$

Calculate the orbit to ECF transformation [ORB2ECEF] using ECEF ephemeris (See the Ancillary Data Preprocessing Algorithm (4.1.4) for this procedure).

Transform precision “delta” vectors to ECEF.

$$[\text{def}] = [\text{ORB2ECEF}][\text{dorb}]$$

$$[\text{dvef}] = [\text{ORB2ECEF}][\text{dvorb}]$$

Adjust ECEF ephemeris by the appropriate “delta” precision vector and store the new ephemeris in the model. These ephemeris points will be used when transforming an input line/sample to an output projection line/sample.

model ECEF position = ephemeris ECEF position + decf
model ECEF velocity = ephemeris ECEF velocity + dvecf

where:
All parameters are 3x1 vectors
ephemeris ecef values are the interpolated one-second ephemeris values in ECEF coordinates

Calculate the orbit to ECI transformation [ORB2ECI] using ECI ephemeris.

Transform precision “delta” vectors to ECI.

$$[deci] = [ORB2ECI][dorb]$$

$$[dveci] = [ORB2ECI][dvorb]$$

Adjust ECI ephemeris by the appropriate “delta” precision vector and store the new ephemeris in the model. These ephemeris points will be used with lunar/stellar observations.

model ECI position = ephemeris ECI position + deci
model ECI velocity = ephemeris ECI velocity + dveci

where:

All parameters are 3x1 vectors
ephemeris eci values are the interpolated one-second ECI ephemeris

Move Satellite Sub-Algorithm

This function computes the satellite position and velocity at a delta time from the ephemeris reference time using Lagrange interpolation. This is a utility sub-algorithm that accesses the model ephemeris data to provide the TIRS position and velocity at any specified time. Since the model ephemeris arrays are inputs to this sub-algorithm it will work with either the ECI or ECEF ephemeris data.

Table 4-36 below summarizes the contents of the TIRS LOS model structure. The estimated size of this structure is approximately 1.5 megabytes.

LOS Model Structure Contents
Satellite Number (8/9)
Format Version Number (for documentation and backward compatibility)
WRS Path
WRS Row (may be fractional)
Acquisition Type (Earth, Lunar, Stellar)
Earth Orientation Parameters
UT1UTC Correction (in seconds)
Pole Wander X Correction (in arc seconds)
Pole Wander Y Correction (in arc seconds)
Image Model
Number of image lines
Image UTC epoch: image_year, image_day, image_seconds
For each line: frame time offset (in seconds) from image epoch
For each line: roll, pitch, yaw high frequency jitter correction (in radians)
Nominal alignment fill table (from CPF) one value per band per SCA (in pixels)
Detector alignment fill table (from L0R/L1R) one value per detector (in pixels)
Sensor Model

LOS Model Structure Contents
TIRS to ACS reference alignment matrix [3x3]
Spacecraft center of mass to TIRS offset in ACS reference frame [3x1] in meters
Integration Time in seconds
Computed Sample Time in seconds
Number of SCAs (3)
Number of Bands (4)
Along-Track IFOV in radians
Across-Track IFOVs (MS and pan) in radians
Number of Detectors per SCA in each Band (4x1 array)
Focal plane model parameters (Legendre coefs) [NSCAxNBANDx2x4] (in radians)
Detector delay table [NSCAxNBANDx2xNDET] (in pixels)
Scene Select Mirror Model
Telescope to SSM alignment matrix [3x3]
Number of SSM encoder angles
Time from image epoch (one per sample, nominally 20 Hz) (in seconds)
SSM angle (one per sample) (in radians)
Ephemeris Model
Scene ephemeris data UTC epoch: imgeph_year, imgeph_day, imgeph_seconds
Number of ephemeris samples
Time from epoch (one per sample, nominally 1 Hz) (in seconds)
Original ECI position estimate (X, Y, Z) (one set per sample) (in meters)
Original ECI velocity estimate (Vx, Vy, Vz) (one set per sample) (in meters/sec)
Original ECEF position estimate (X, Y, Z) (one set per sample) (in meters)
Original ECEF velocity estimate (Vx, Vy, Vz) (one set per sample) (in meters/sec)
Corrected ECI position estimate (X, Y, Z) (one set per sample) (in meters)
Corrected ECI velocity estimate (Vx, Vy, Vz) (one set per sample) (in meters/sec)
Corrected ECEF position estimate (X, Y, Z) (one set per sample) (in meters)
Corrected ECEF velocity estimate (Vx, Vy, Vz) (one set per sample) (in meters/sec)
Attitude Model
Scene attitude data UTC epoch: imgatt_year, imgatt_day, imgatt_seconds
Number of attitude samples
Time from epoch (one per sample, nominally 50 Hz) (in seconds)
Original Roll, pitch, yaw estimate (one per sample) (in radians)
Corrected Roll, pitch, yaw estimate (one per sample) (in radians)
Precision Correction Model
Precision reference time (t_ref) seconds from image epoch
Ephemeris correction order: eph_order (0 none, 1 for Earth-view and lunar/stellar)
X correction model: x_bias, x_rate (meters, meters/sec)
Y correction model: y_bias, y_rate (meters, meters/sec)
Z correction model: z_bias, z_rate (meters, meters/sec)
Attitude correction order: att_order (0 none, 1 for Earth, 2 for lunar/stellar)
Roll correction model: roll_bias, roll_rate, roll_acc (rad, rad/sec, rad/sec^2)

LOS Model Structure Contents
Pitch correction model: pitch_bias, pitch_rate, pitch_acc (rad, rad/sec, rad/sec^2)
Yaw correction model: yaw_bias, yaw_rate, yaw_acc (rad, rad/sec, rad/sec^2)

Table 4-36. TIRS LOS Model Structure Contents

Note that in the precision correction model only the correction model array elements up to att_order are valid. For example, for Earth-view scenes att_order = 1 and roll_corr[0] = roll_bias, roll_corr[1] = roll_rate and roll_corr[2] is not used.

4.3.1.7 Notes

Some additional background assumptions and notes include the following:

1. The static precession, nutation, and sidereal time parameters needed to convert Earth-Centered Inertial J2000 to/from WGS84 Earth-Fixed are built into the software rather than being provided as input data. The dynamic terms (UT1UTC correction, polar wander) are provided in the CPF. For Landsat 8/9, the CPF was expanded to include a leap second table to allow for converting spacecraft TAI-reference time codes to UTC. This TAI to UTC time conversion and the ECI/ECEF conversion algorithm are discussed in the ancillary data preprocessing algorithm description document (4.1.4).
2. While it seems to be generally agreed that the TIRS Level 0R/Level 1R data will not use fill pixels to nominally align bands and/or SCAs, it may include fill pixels to achieve nominal detector-to-detector alignment in the case where bad detectors are replaced from the redundant detector row. Since this is an existing capability in the heritage OLI logic, the L0R/L1R detector alignment fill table input identified in the input table will be retained as the mechanism for the Level 0R/1R data to identify the number of any fill pixels used. In practice, it may be preferable to keep the TIRS L0R in strict time order (with no fill) even if dead detector replacement is performed.
3. The "thresholds and limits" parameters, stored either in system tables or the database for L7 and ALI, will be included in the CPF for Landsat 8/9. This will make date specific changes, e.g., due to a change in the nominal orbit during early- or late-mission operations, easier to manage.
4. The current algorithm baseline is to use the heritage attitude model roll-pitch-yaw representation. This could be updated in a future revision to use a quaternion representation. This is the motivation for including both quaternion and roll-pitch-yaw representations of the attitude data sequence in the output from the ancillary data preprocessing algorithm (4.1.4).
5. This algorithm includes a simple image time code validation/smoother function to fix errors and/or smooth out quantization effects in the downlinked time codes. This may not be necessary or it may need to be more elaborate depending on the reliability of the TIRS time codes.
6. The baseline algorithm prototype implementation allows the precision correction model parameters to be provided as optional input parameters. This would not be used for operational data processing and these parameters would not ordinarily be provided, with their values defaulting to those set in the Initialize the Precision

Model sub-algorithm. Having such an R&D capability to force model corrections at model creation time can prove useful in applications such as data simulation and anomaly resolution.

7. The reliability of the SSM encoder telemetry was unknown at the time of algorithm development, although on-orbit TIRS telemetry has exhibited good performance. As a contingency, the baseline model includes a quality check and smoothing logic to allow for SSM encoder data preprocessing. The quality check is based on a threshold check that ensures sample-to-sample consistency. The threshold is a CPF parameter. A check for consistency with a nominal value may also be required.
8. The baseline model does not include any temperature dependent effects in either the SSM or in the overall TIRS alignment. The temperature telemetry provided in the TIRS ancillary data would provide a means for investigating any such dependencies on orbit. This algorithm assumes that the TIRS temperature telemetry will be collected and trended by the radiometric processing algorithms and would therefore be available, if needed, for future implementation of temperature-based calibration adjustments.
9. The SSM encoder position is provided at a 20 Hz sampling rate even though the SSM telemetry packets are generated at 1 Hz. Each packet contains 20 samples, with each sample representing one 24-bit encoder read out.
10. The TIRS detector deselect mechanism and detector offset geometry differ from the OLI versions but the same correction logic can be applied. In the OLI case, adjacent redundant detectors are switched on in place of the defective primary detectors causing the active detector location to be shifted in the along-track direction. This shift is in addition to the normal even/odd detector offset. For TIRS, there is no even/odd offset and instead of having individual redundant detectors that must be switched on individually, an entire redundant row of detectors is downlinked for each band. Detector replacement is performed in Level 0 processing where the samples from defective primary detectors are swapped with the samples from the corresponding detector in the redundant row. The net result is that TIRS detectors will have an integer detector offset of either 0, for detectors from the primary row, or whatever the line offset happens to be between the primary and redundant detector rows, for detectors that are replaced/deselected. Since the selection of primary and redundant rows will be made separately for each band on each SCA, these offsets can vary from SCA-to-SCA but will be constant within a given band on a given SCA.
    Analysis of the TIRS optical model has shown that the 2nd order Legendre polynomial model used to generate OLI lines-of-sight will not be adequate for TIRS. This is due primarily to the larger field of view of the TIRS SCAs. Since each SCA covers a larger portion of the instrument's field of view, it is subject to more of the optical distortion variations that occur across the field of view. Initial analysis indicates that a 3rd order Legendre model will capture the nominal TIRS detector lines of sight for each band and each SCA with sufficient fidelity. The TIRS detector line-of-sight model has been updated accordingly.

4.3.2 TIRS Line-of-Sight Projection/Grid Generation

4.3.2.1 Background/Introduction

The LOS projection and grid generation algorithm uses the TIRS LOS model, created by the TIRS LOS model creation algorithm, to calculate the intersection of the projected lines-of-sight from selected TIRS detector samples (pixels) with an Earth model (WGS84). The spacecraft position and pointing, TIRS instrument alignment and offset information, TIRS scene SSM angle, and image timing data contained in the LOS model are used to construct the LOS for an individual TIRS detector at a particular sample time. We then calculate the location where that line of sight intersects the Earth's surface, as defined by the WGS84 Earth ellipsoid or a specified elevation above or below that ellipsoid. LOS intersections for an array of detector samples that span each TIRS SCA and spectral band are computed at the WGS84 ellipsoid surface as well as at a range of elevation levels selected to span the actual terrain elevations found in the image area. The resulting array of projected lines-of-sight forms a three-dimensional grid of input (Level 1R) image pixel line/sample to output space (Level 1G) mappings that can be used to interpolate input/output pixel mappings for intermediate points. The resulting ability to rapidly compute input/output mappings greatly facilitates image resampling.

The TIRS LOS projection and grid generation algorithm can also work in an "inertial direction" mode in which the output space is in angular units with respect to a set of reference inertial directions. This mode is used to process lunar data wherein the inertial coordinates (declination and right ascension) of the moon, computed from a planetary ephemeris, are used as the reference to define the output image frame. In this case the lines-of-sight are computed in inertial coordinates but are not projected to the Earth's surface. This mode of operation is not specifically required for TIRS imagery, but the capability is retained in this algorithm to maintain compatibility with the corresponding OLI algorithm, to facilitate future convergence.

Concerns about the temporal (line direction) grid density that would be required to adequately capture attitude deviations (jitter) at frequencies above 10 Hz motivated the addition of new grid functionality to support high frequency image correction at image resampling time. Specifically, jitter sensitivity coefficients were added to each grid cell to allow the high frequency attitude data in the TIRS line-of-sight model jitter table to be converted to corresponding input image space line/sample offsets. These coefficients are used by the resampler to compute high frequency line/sample corrections that refine the output-to-input space image coordinate mappings provided by the grid. This allows the grid to model only lower frequency effects making a sparser grid sampling in the time (line) direction possible.

Due to the layout of the TIRS focal plane, there is an along-track offset between the spectral bands within each SCA, an along-track offset between the outboard (odd) and inboard (even) SCAs, and a reversal of the band ordering in adjacent SCAs. This leads to an along-track offset in the imagery coverage area for a given band between odd and even SCAs as well as an offset between bands within each SCA. To create more

uniform image coverage within a geometrically corrected output product, the leading and trailing imagery associated with these offsets is trimmed (at image resampling time) based on image active area bounds stored in the grid.

The TIRS LOS projection and grid generation algorithm is derived from the corresponding OLI algorithm. Its implementation is very similar to the oligrid application.

4.3.2.2 Dependencies

The TIRS LOS projection and grid generation algorithm assumes that the TIRS LOS model creation algorithm has been executed to construct and store the TIRS LOS model.

4.3.2.3 Inputs

The TIRS LOS projection and grid generation algorithm and its component sub-algorithms use the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs
ODL File (implementation)
CPF File Name
TIRS LOS Model File Name
DEM File Name
Reference Grid File (optional) (for transferring framing parameters)
Reference Band (optional) (reference grid band to use)
Output Image Framing Parameters:
WRS Path for path-oriented scene framing (not necessarily the LOS model path)
WRS Row for path-oriented scene framing (not necessarily the LOS model row)
Map Projection (UTM, SOM, PS)
UTM Zone (use 0 to have code compute the zone)
Map Projection Parameters
Output Pixel Size(s)
Output Image Orientation
Frame Type (e.g., MINBOX)
Frame Bounds (e.g., corner coordinates, image size)
Grid Options:
Bands to Grid
CPF file contents
Maximum detector offset for each band
Thresholds and Limits (replaces System Table)
Grid Density (line/sample/height)
Default (WGS84) Spheroid and Datum Codes
TIRS LOS Model file contents (see TIRS LOS Model Creation Algorithm for details)
WGS84 Earth Model Parameters
Earth Angular Velocity (rotation rate) in radians/second
Speed of light (in meters/second)
Acquisition Type (Earth, Lunar, Stellar)
TIRS to ACS reference alignment matrix
Spacecraft CM to TIRS offset in ACS reference frame (new)

Algorithm Inputs
SSM model (Telescope alignment matrix and time-indexed SSM angles)
Focal plane model parameters (Legendre coefs)
Detector delay table
Smoothed ephemeris at 1 second intervals (original and corrected)
Low-pass filtered attitude history (original and corrected)
High frequency attitude perturbations (roll, pitch, yaw) per image line (jitter table)
Image time codes
Integration Time
Nominal detector alignment fill table
L0R detector alignment Fill Table
DEM file contents
Min and Max Elevation
NOVAS Planetary Ephemeris file contents (Note: The NOVAS ephemeris file name is provided via an environment variable.)
JPL Ephemeris Table (DE421) for celestial bodies (i.e., the moon) (see note 1)

4.3.2.4 Outputs

TIRS Grid (see Table 4-37 and Table 4-38 below for detailed grid structure contents)
Grid Header (WRS path/row, acquisition type)
Output Image Framing Information (corner coordinates, map projection)
Image active area latitude/longitude bounds (for each band)
Grid Structure Information (number of bands/SCAs)
Grid Structures (one per SCA, per band)
Band number
Image dimensions (line/sample)
Pixel size
Grid cell size (image lines/samples per cell)
Grid dimensions (# rows/# columns/# Z-planes)
Z-plane zero reference and height increment
Arrays of input line/sample grid point coordinates
Arrays of output line and sample grid point mappings
Arrays of even/odd offset coefficients (2 per grid cell)
Arrays of forward (input/output) mapping polynomials (8 per grid cell per Z-plane)
Arrays of inverse (output/input) mapping polynomials (8 per grid cell per Z-plane)
Arrays of roll-pitch-yaw jitter line sensitivity coefficients (3 per grid cell per Z-plane)
Arrays of roll-pitch-yaw jitter sample sensitivity coefficients (3 per grid cell per Z-plane)
Rough mapping polynomials (one set per Z-plane)

4.3.2.5 Options

A NOVAS planetary ephemeris file (e.g., JPL DE421) must be provided when the Acquisition Type (in the LOS model) is Lunar.

4.3.2.6 Procedure

The LOS Projection algorithm uses the TIRS LOS model created by the TIRS LOS Model Creation algorithm to relate TIRS image pixels to ground locations or, in the case of lunar/stellar images, to ECI directions. The LOS model contains several components including: Earth orientation parameters, an image model (validated image time codes),

a sensor model (including SSM angles), an ephemeris model, and an attitude model. The Level 1R image line/sample location is used to compute a time of observation (from the image model), a LOS vector (from the sensor model), the spacecraft position (from the ephemeris model) at the time of observation, and the spacecraft attitude (from the attitude model) at the time of observation. The LOS vector is projected to the Earth's surface, either the topographic surface at a specified elevation (e.g., derived from an input Digital Elevation Model), or the WGS84 ellipsoid surface, to compute the ground position associated with that Level 1R image location. This LOS projection procedure relating an input image location to an output ground location is referred to as the forward model. In image resampling, we typically need to find the Level 1R input space line/sample location corresponding to a particular Level 1G output space location so that the corresponding image intensity can be interpolated from the Level 1R data. This "inverse model" computation must be performed for every pixel in the output Level 1G product. To make this computation efficient, we create a table, or grid, of input/output mappings, parameterized by height, for use by the TIRS image resampling algorithm. Both the forward model and grid generation procedures are described in this algorithm description document.

The Geometric Grid

The geometric grid provides a mapping from input Level 1R line/sample space to output Level 1G line/sample space. As such, it incorporates not only the sensor LOS to Earth intersection geometry captured by the forward model, but also the output image framing information, such as scene corners, map projection, pixel size, image orientation, and the bounds of the active image area for each band. The gridding procedure generates a mapping grid that defines a transformation from the instrument perspective (input space) to a user specified output projection on the ground (output space). This output frame may be map-oriented (north-up) or path-oriented for Earth-view acquisitions. Alternatively, the user may specify a previously generated grid file such that this grid’s scene framing information is used for the generation of the new grid. Celestial (lunar/stellar) acquisitions use an output frame based on inertial right ascension and declination coordinates. Once the frame is determined in output space, the input space is gridded. Then the grid in input space is mapped to the output space using the forward model. Transformation coefficients to transform a grid cell from input to output space are determined, as well as coefficients to transform a grid cell from output to input space.

The concept behind creating this resampling grid is to define only a sparse set of points for the relationship between an input line and sample location to output line and sample location (see Figure 4-62). Four grid points define a grid cell. A grid cell is defined as a rectangle in input space but will be distorted when mapped to the output space. The sampling of points between grid cell points is chosen such that any two points defining a grid cell and a line in input space will map to a line in output space. Therefore every grid cell defines a bilinear mapping between the input and output space and vice versa. The method of only mapping and storing a small set of input points is much more efficient than trying to map points individually by invoking the LOS model for each point. This is

especially the case since a rigorous implementation of the inverse model would have to be iterative.

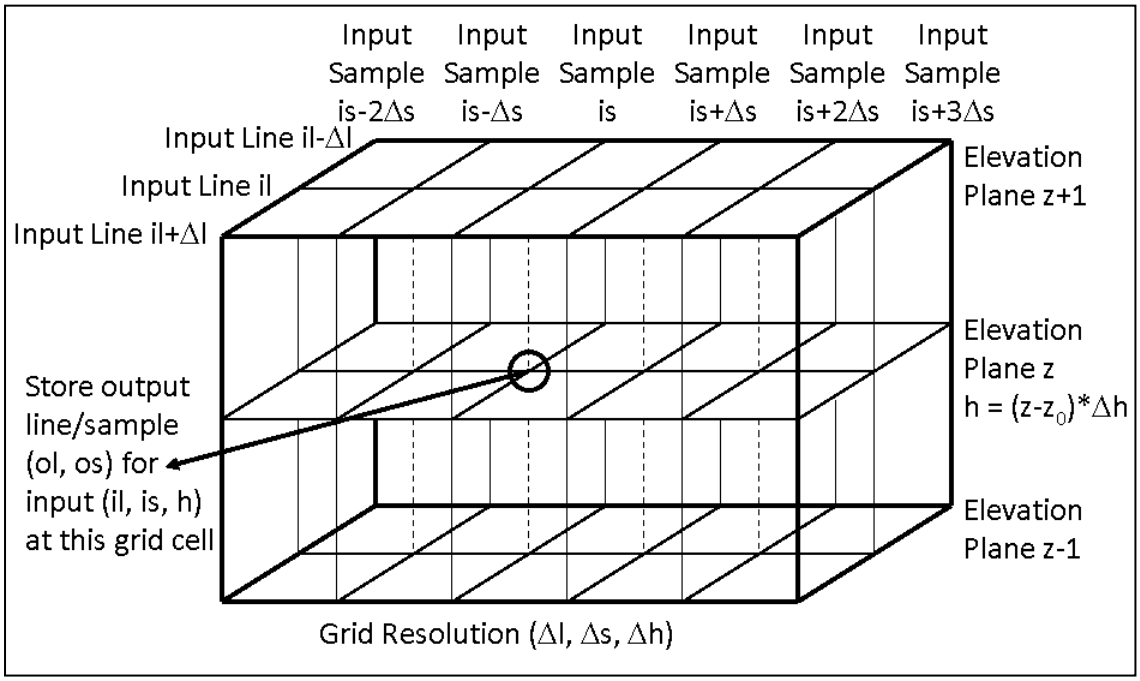

Figure 4-62. 3D Grid Structure

The 3D grid structure stores the output space line/sample coordinates corresponding to an array of input space line/sample/height coordinates.

The LOS projection grid contains projection information and three groups of mapping coefficients—one for mapping each grid cell from output space to input space (inverse), a second for mapping each grid cell from input space to output space (forward), and third that gives an approximation or “rough” mapping of output space to input space. The first two mappings are described by a set of bilinear polynomials. The input space is represented by a line and sample location while the output space is represented by a line and sample location along with a Z component, where Z represents elevation. The output lines and samples can in turn be converted to X, Y projection space location by using the output image’s upper-left projection coordinate and pixel size information in the grid header. Figure 4-63 shows how one input grid cell is mapped to a number of output grid cells, each grid cell representing a different elevation.

The number of grid cells is dependent on the line and sample size of each grid cell in the input image, elevation maximum, elevation minimum, and elevation increment. The input space is made up of evenly spaced samples and lines, values are associated with integer locations and can be indexed by an array of values: input_line[row] and input_sample[column]. Row refers to the index number, or row number, associated with the line spacing while column refers to the index number, or column number, associated with the sample spacing. The output lines and samples typically do not fall on integer

values (see Figure 4-63). This creates a two dimensional array of indices for output line and sample locations. Adding elevation indices produces a three dimensional array for output line and sample locations. The output lines and samples are then indexed by output_line[z][row][column] and output_sample[z][row][column] where Z refers to an elevation value. The row and column are the indices associated with the gridding of the raw input space. Since there is a mapping polynomial for each grid cell, the mapping polynomial coefficients are indexed by the same method as that used for output lines and samples; i.e., there are z*row*column sets of mapping coefficients.

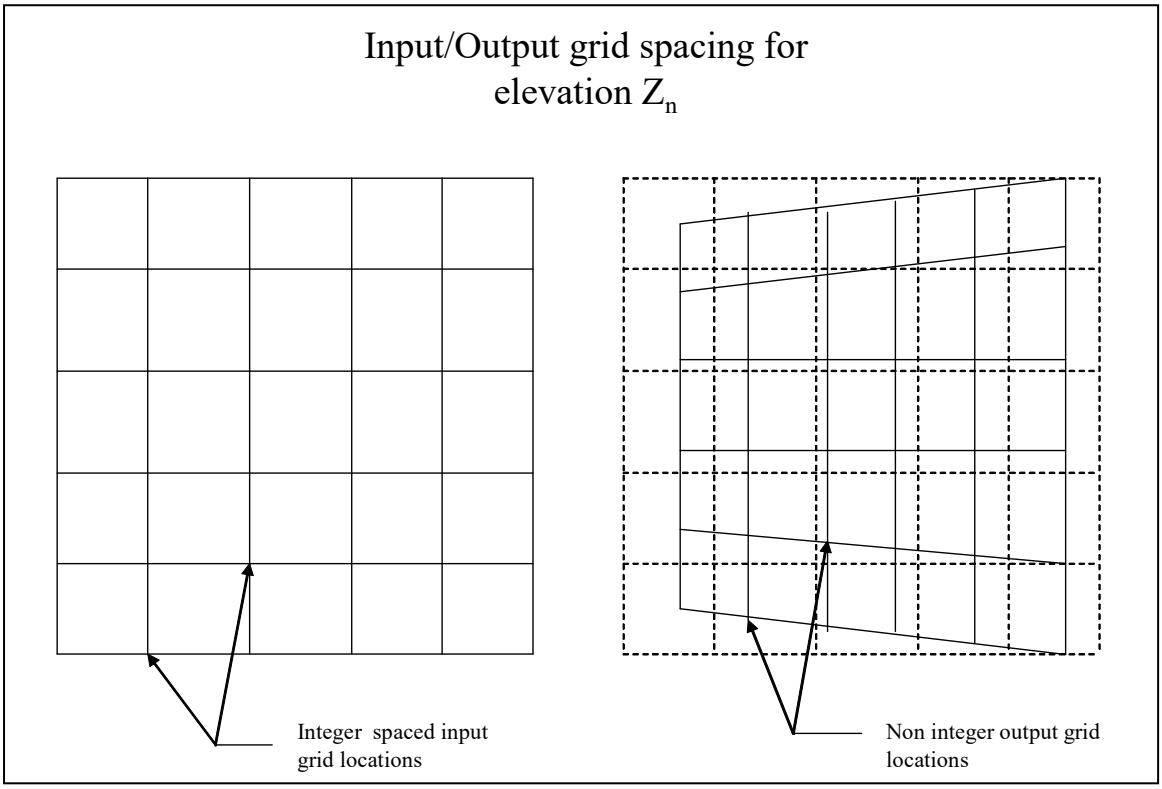

Figure 4-63. Mapping Integer Locations to "Non-integer" Locations

If a grid is being generated for a non-terrain corrected image (i.e., no correction for relief is being applied) then the index for z is set such that $z_{elev=0}$ = zero elevation. Note that $z_{elev=0}$ does not necessarily have to be the first index in the array since there could be values for negative elevations. If the grid is being generated for a terrain corrected image, then the indexes z_n and z_{n+1} are used such that the elevation belonging to the output location falls between the elevations associated with the indexes n and n+1. When performing an inverse mapping for a terrain corrected image, two sets of input lines and samples are calculated from the polynomials for n and n+1. The actual input line and sample is interpolated between these lines and samples.

Example:

Output line/sample has r = row, c = col and z=n, n+1. If the inverse mapping coefficients are *a* and *b* for line and sample respectively then:

input_line$_n$ = *bilinear*(a_n,output_line,output_sample)
input_sample$_n$ = *bilinear*(b_n,output_line,output_sample)
input_line$_{n+1}$ = *bilinear*(a_{n+1},output_line,output_sample)
input_sample$_{n+1}$ = *bilinear*(b_{n+1},output_line,output_sample)

bilinear is the bilinear mapping function (described below) for each grid cell.

If *e* is the elevation for the output line and sample location then the weights used to interpolate between the two input line/sample locations are as follows:

$$w_n = \frac{e_{n+1} - e}{e_{n+1} - e_n} \qquad w_{n+1} = \frac{e - e_n}{e_{n+1} - e_n}$$

e_n, e_{n+1} and *e* are the elevations associated with z_n , z_{n+1} , and the output line and sample respectively.

The final line/sample location is found from:

input_line = w_n * input_line$_n$ + w_{n+1} * input_line$_{n+1}$
input_sample = w_n * input_sample$_n$ + w_{n+1} * input_sample$_{n+1}$

The grid must contain a zero elevation plane. If the input minimum elevation is greater than zero it is set to zero. If the input maximum elevation is less than zero it is set to zero.

Given the elevation maximum, minimum, and increment determine the number of z planes and the index of the zero elevation plane. Adjust the minimum and maximum elevations to be consistent with the elevation increment.

The number of z planes is determined from:

$$\text{number of z planes} = (\text{int})\left(\text{ceil}\left(\frac{\text{elevation maximum}}{\text{elevation increment}} \right) - \text{floor}\left(\frac{\text{elevation minimum}}{\text{elevation increment}} \right) \right) + 1$$

The subsequent grid interpolation logic assumes that there are at least 2 z planes so the number of z planes is set to 2 if the calculation above results in fewer than 2 planes. This can only happen if the minimum and maximum elevations are both zero.

The plane for an elevation of zero is then found at:

$$z_{elev=0} = -\text{floor}\left(\frac{\text{elevation minimum}}{\text{elevation increment}} \right)$$

The new minimum and maximum elevation due to the values calculated above are as follows:

$$\text{elevation minimum} = -z_{elev=0} * (\text{elevation increment})$$

$$\text{elevation maximum} = (\text{number of } z - 1 - z_{elev=0}) * \text{elevation increment}$$

LOS Projection/Grid Generation Procedure Overview

The LOS Projection/Grid Generation procedure is executed in five stages:

1. Data Input - First, the required inputs are loaded. This includes reading the processing parameters from the input ODL parameter file, loading the TIRS LOS model from its HDF file, reading static gridding parameters from the CPF, and loading the elevation data from the DEM.
2. Scene Framing - The parameters of the output image space are computed based on the scene framing scheme specified in the input ODL file or are loaded from a previously generated grid file. This includes calculating bounds for the active image area that excludes the leading and trailing SCA imagery, and using one of several available methods for determining the Level 1G scene corners. The scene framing parameters are stored in the grid structure for eventual inclusion in the geometric metadata for the Level 1G product.
3. Grid Definition - The grid parameters are established to ensure adequate density in the space (sample), time (line), and elevation (z-plane) dimensions. The required data structures are allocated and initialized.
4. Grid Construction - The forward model is invoked for each grid intersection to construct the array of input space to output space mappings. A separate grid structure is created for each SCA and each band. The grid mapping polynomial coefficients are computed from the input space to output space mapping results for each grid cell. Once the basic grid mappings are defined, the forward model is invoked with small attitude perturbations about each axis in order to evaluate the sensitivity of the input space to output space mapping to small attitude deviations. The resulting sensitivity coefficients are stored with each grid cell for subsequent use in computing high frequency jitter corrections during image resampling. Figure 4-64 shows a data flow for the creation and use of these new coefficients.
5. Finalize and Output Grid - Derived grid parameters such as the global rough mapping coefficients, are added to the grid structure, and the entire structure is written to a disk file. This also includes evaluating the small, but significant, parallax effects caused by the time delay between adjacent primary and (replaced) redundant detectors as they sample the same along-track location. These effects are modeled in the grid as along- and across-track sensitivity coefficients that are scaled by the output point elevation and the even/odd detector offset, which can vary by pixel for TIRS (due to detector deselect/replacement). This parallax effect is not as pervasive in TIRS as compared to OLI since the primary TIRS detectors are not arranged with

even/odd detector stagger as is the case for OLI. The parallax correction is retained in the TIRS grid to account for bad detector replacement (when a primary row detector is replaced by the corresponding redundant row detector) and to maintain compatibility with the OLI grid.

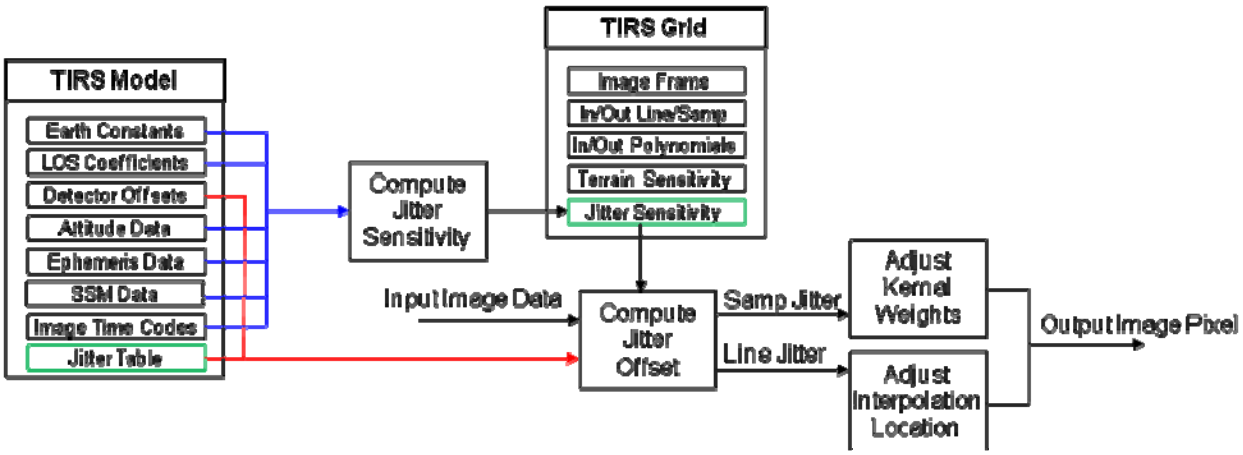

Figure 4-64. Jitter Correction Data Flow

Figure 4-65 shows a block diagram for the TIRS LOS Projection algorithm.

4.3.2.6.1 Stage 1 - Data Input

The data input stage involves loading the information required to perform grid processing. This includes reading the framing parameters for the output scene from the ODL file, reading grid structural parameters from the CPF, loading the TIRS LOS model structure in preparation for invoking the forward model, and reading the DEM to determine the elevation range for the image.

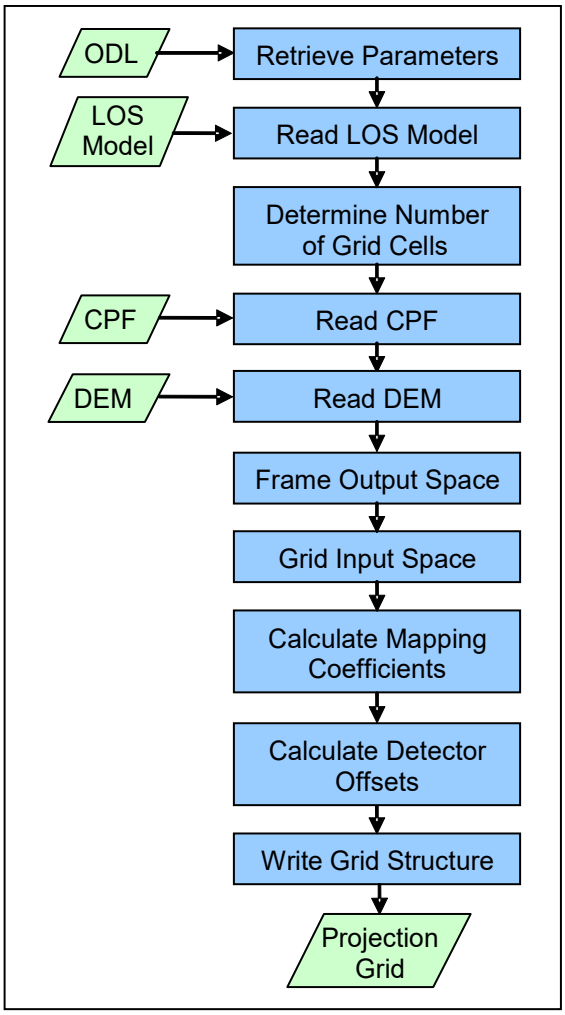

Figure 4-65. Line-of-Sight Projection Block Diagram

4.3.2.6.2 Stage 2 - Scene Framing

Framing the output image space involves determining the geographic extent of the output image to be generated by the resampler. This geographic extent of the output image space is referred to as the output space “frame,” and is specified in output image projection coordinates. There are five different methods that are used to determine the output frame for Earth-viewing acquisitions. Note that the fifth method is new for TIRS and is not really a “framing method” in the same sense. Logic that supports scene framing for celestial (lunar and stellar) scenes using a maximum bounding rectangle (maxbox) approach based on inertial LOS declination and right ascension coordinates, is retained in the code reused from the corresponding OLI algorithm, but this capability is not required for TIRS so it is only described briefly below. These methods use the calculated coverage bounds of each band/SCA in different ways, with some excluding the leading and trailing SCA imagery based on a calculated active image area, and some including the leading/trailing imagery so as to preserve all available input pixels (e.g., for calibration purposes). Thus, the calculation of the active image area for each band is the first step in scene framing.

Calculating the Active Image Area

The along-track offsets between spectral bands and even/odd SCAs create an uneven coverage pattern when projected into output image space. To provide a more regular output image coverage boundary, we define a rectangular active image area that excludes the excess trailing imagery from even SCAs and the excess leading imagery from odd SCAs. This active area is used for the minbox framing methods, which seek to limit the output product area to provide consistent, contiguous coverage, but are ignored for maxbox framing methods, where all available imagery is desired.

The active image area is computed by constructing 8 critical SCA corner points, labeled C1 through C8 in Figure 4-66 below. This figure depicts the current understanding of the TIRS field of view orientation with respect to object space, but the algorithm described here will work so long as the SCAs are numbered sequentially across the field of view, in either direction. Points C1 and C2 define the top edge of the active area, C3 and C4 the right edge, C5 and C6 the bottom edge, and C7 and C8 the left edge. Note that points C4 and C5 are the same (the lower-right corner of SCA01) as are points C6 and C7 (the lower-left corner of SCA03). The forward model projects these 8 line/sample locations to object space, computing the latitude/longitude coordinates of the WGS84 ellipsoid intersection for each point.

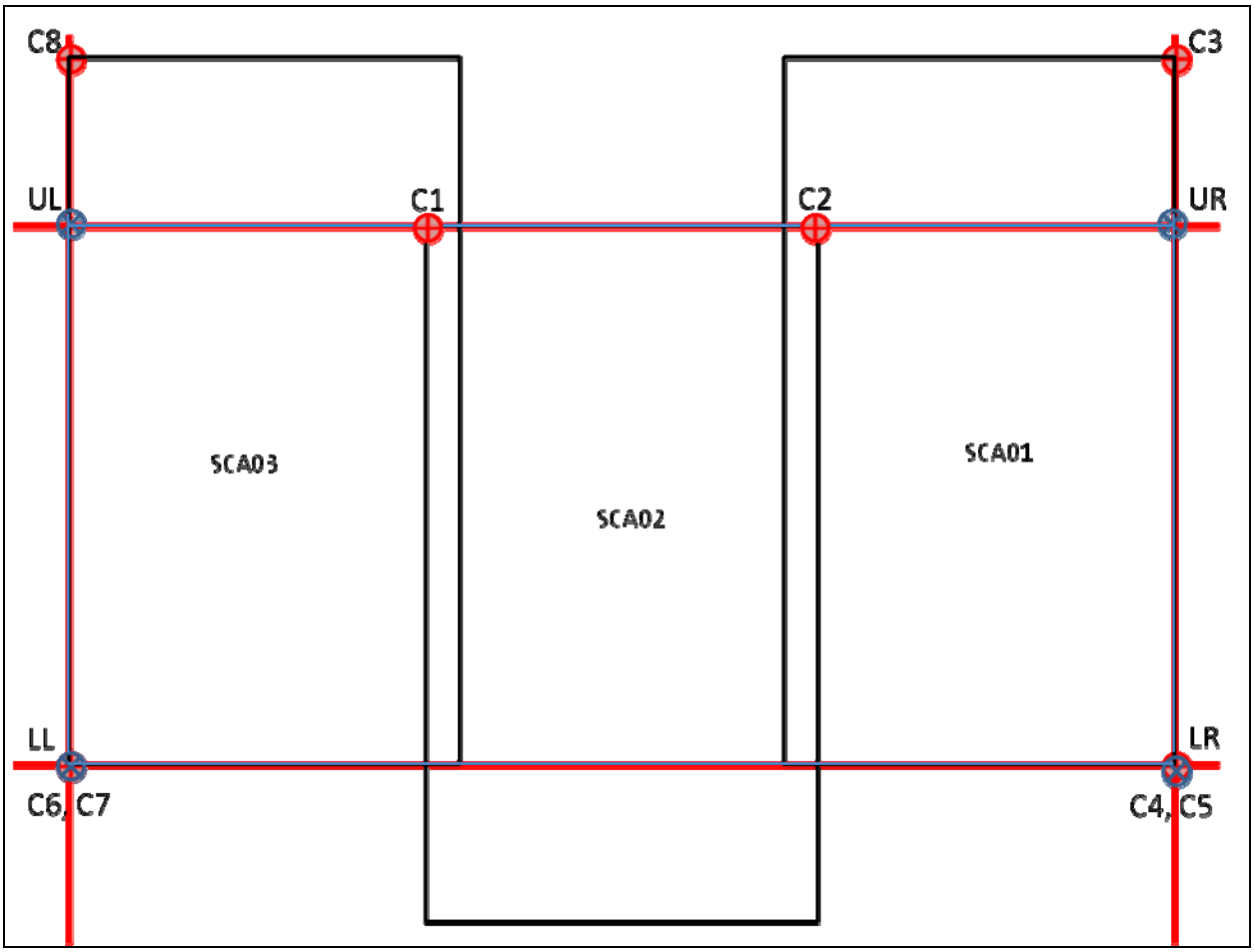

Figure 4-66. Active Image Area Construction

The corner point assignments are made automatically by examining the SCA across-track and along-track Legendre coefficients to determine: 1) whether SCA01 is on the left (+Y) or right (-Y) side of the scene; 2) whether even or odd SCAs lead; and 3) whether the sample number increases in the –Y or +Y direction. If the across-track Legendre constant term (coef_y0) for SCA01 is positive then it is the left-most SCA and SCA03 is the right-most. If the along-track Legendre constant term (coef_x0) for SCA01 is greater than that for SCA02, then the odd SCAs lead. If the across-track Legendre linear term (coef_y1) for SCA01 is negative, then the sample number increases in the –Y direction.

Having determined the orientation of the SCAs, we assign the top edge to the left-most leading SCA UL corner and the right-most leading SCA UR corner, the right edge to the right-most SCA UR and LR corners, the bottom edge to the right-most trailing SCA LR corner and left-most trailing SCA LL corner, and the left edge to the left-most SCA LL and UL corners. As shown in the figure, for the TIRS: C1 = SCA02 (left-most leading SCA) UL, C2 = SCA02 (right-most leading SCA) UR, C3 = SCA01 (right-most SCA) UR, C4 = SCA01 (right-most SCA) LR, C5 = SCA01 (right-most trailing SCA) LR, C6 = SCA03 (left-most trailing SCA) LL, C7 = SCA03 (left-most SCA) LL, and C8 = SCA03

(left-most SCA) UL. Note that these assignments are based on the current TIRS SCA ordering of SCA-B = SCA01, SCA-C = SCA02, and SCA-A = SCA03, and could change if the SCA numbering system is revised. If this were to happen, the change would be reflected in the Legendre coefficients, so the logic described here would automatically compensate.

The geodetic latitudes computed by the forward model are converted to geocentric latitudes using:

$\theta = \arctan((1-e^2) \tan(\phi))$

where:

θ = geocentric latitude

ϕ = geodetic latitude

e^2 = WGS84 ellipsoid eccentricity squared

This creates a set of 8 geocentric latitude/longitude (θ_i, λ_i) pairs, one for each “critical” corner, noting that geocentric longitude is equal to geodetic longitude.

Use the geocentric latitude/longitude to construct a geocentric unit vector for each corner:

$$X_i = \begin{bmatrix} \cos(\lambda_i)\cos(\theta_i) \\ \sin(\lambda_i)\cos(\theta_i) \\ \sin(\theta_i) \end{bmatrix}$$

Note that these vectors are inherently normalized.

Construct vectors normal to the top, right, bottom, and left edge great circles by taking cross products of the corner vectors:

$$X_T = \frac{X_1 \times X_2}{|X_1 \times X_2|} \qquad X_R = \frac{X_3 \times X_4}{|X_3 \times X_4|} \qquad X_B = \frac{X_5 \times X_6}{|X_5 \times X_6|} \qquad X_L = \frac{X_7 \times X_8}{|X_7 \times X_8|}$$

Construct corner vectors from the edge vectors:

$$X_{UL} = \frac{X_T \times X_L}{|X_T \times X_L|} \qquad X_{UR} = \frac{X_R \times X_T}{|X_R \times X_T|} \qquad X_{LL} = \frac{X_L \times X_B}{|X_L \times X_B|} \qquad X_{LR} = \frac{X_B \times X_R}{|X_B \times X_R|}$$

The top and bottom edges are next checked against all the SCA corners to ensure that any curvature in the SCA field angle pattern is accounted for. This is done to suppress residual SCA edge “raggedness.”

Adjust the top edge:

Construct a vector in the plane of the top edge great circle:

$$X_g = \frac{(X_{UR} - X_{UL}) \times X_T}{|(X_{UR} - X_{UL}) \times X_T|}$$

Initialize the minimum "out of plane" distance: $a_{min} = 1$
For each SCA:
For the two upper corners: UL (0,0) and UR (ns-1,0):
Use the forward model to project the corner.
Convert the geodetic latitude to geocentric latitude as above.
Construct a geocentric unit vector, X_i, as above.
Project the unit vector onto the X_g and X_T vectors and compute the ratio:

$$a_i = \frac{X_i \bullet X_T}{X_i \bullet X_g}$$

If $a_i < a_{min}$
$a_{min} = a_i$
$X_{min} = X_i$
Next corner
Next SCA

If $a_{min} < 0$ then the innermost corner lies inside the current active area and we need to adjust the top edge:

$$X'_g = \frac{(X_{\min} \bullet X_T)X_T + (X_{\min} \bullet X_g)X_g}{\left|(X_{\min} \bullet X_T)X_T + (X_{\min} \bullet X_g)X_g\right|}$$

$$X'_T = \frac{X'_g \times (X_{UR} - X_{UL})}{\left|X'_g \times (X_{UR} - X_{UL})\right|}$$

And update the top corner vectors using the adjusted edge vectors:

$$X_{UL} = \frac{X'_T \times X_L}{\left|X'_T \times X_L\right|} \qquad X_{UR} = \frac{X_R \times X'_T}{\left|X_R \times X'_T\right|}$$

Adjust the bottom edge:
Construct a vector in the plane of the bottom edge great circle:

$$X_g = \frac{(X_{LL} - X_{LR}) \times X_B}{\left|(X_{LL} - X_{LR}) \times X_B\right|}$$

Initialize the minimum "out of plane" distance: $a_{min} = 1$
For each SCA:
For the two lower corners: LL (0,nl-1) and LR (ns-1,nl-1):
Use the forward model to project the corner.
Convert the geodetic latitude to geocentric latitude as above.
Construct a geocentric unit vector, X_i, as above.
Project the unit vector onto the X_g and X_B vectors and compute the ratio:

$$a_i = \frac{X_i \bullet X_B}{X_i \bullet X_g}$$

If $a_i < a_{min}$

$a_{min} = a_i$

$X_{min} = X_i$

Next corner

Next SCA

If $a_{min} < 0$ then the innermost corner lies inside the current active area and we need to adjust the bottom edge:

$$X'_g = \frac{(X_{\min} \bullet X_B)X_B + (X_{\min} \bullet X_g)X_g}{\left|(X_{\min} \bullet X_B)X_B + (X_{\min} \bullet X_g)X_g\right|}$$

$$X'_B = \frac{X'_g \times (X_{LL} - X_{LR})}{\left|X'_g \times (X_{LL} - X_{LR})\right|}$$

And update the bottom corner vectors using the adjusted edge vectors:

$$X_{LL} = \frac{X_L \times X'_B}{\left|X_L \times X'_B\right|} \qquad X_{LR} = \frac{X'_B \times X_R}{\left|X'_B \times X_R\right|}$$

Convert the four corner vectors to the corresponding geodetic latitude/longitude:

λ = atan2(X.y, X.x)

θ = atan2(X.z, $\sqrt{\text{X.x}^2 + \text{X.y}^2}$)

ϕ = atan(tan(θ) / (1-e^2))

The four latitude/longitude corners are the bounds of the active image area.

Once the active image area bounds are calculated, the output product frame is determined using one of the following methods:

Method 1: PROJBOX

The user defines the upper-left and lower-right corner coordinates of the area of interest in target map projection coordinates. These coordinates are then projected to the output projection coordinate system using the Projection Transformation Package (see the Projection Transformation sub-algorithm below). This usually results in a non-rectangular area so a minimum-bounding rectangle is found (in terms of minimum and maximum X and Y projection coordinates) in the resulting output space. This minimum-bounding rectangle defines the output space frame. The output image pixel size is then applied to the projection space to determine the number of lines and samples in the output space. This creates an output image that is map projection north-up.

Method 2: MINBOX

The image active areas for each band, calculated previously, are converted to the specified output map projection coordinate system and used in a minimum bounding rectangle computation to create an output image frame that includes the active area for each band. The computed (latitude/longitude) active area corners are maintained in the grid for subsequent use by the image resampler, so that the output product image will not include leading/trailing SCA imagery.

Method 3: MAXBOX

The four corners of each SCA in each band are projected to the Earth. The maximum and minimum latitude and longitude found across all SCAs and all bands are used to establish the output scene frame in the manner described above for the PROJBOX method. This creates an output frame that contains all input pixels from all bands. The previously calculated image active areas are ignored in this process, and the band active area corners are all set equal to the output product corners. Leading and trailing SCA imagery is thereby not excluded from MAXBOX framed products.

Method 4: PATH

The user specifies a path-oriented Landsat product in either the SOM or UTM projection. In this case, the framing coordinates are not user-specified. For a standard PATH scene, the frame is a preset number of lines and samples based on the Landsat WRS scene size and the maximum rotation needed to create a path-oriented product. For PATH_MAXBOX scenes the MAXBOX logic described above is applied to the path-oriented scene to ensure that the output frame contains all input pixels from all bands. For PATH_MINBOX the MINBOX logic described above is applied to the path-oriented scene so that the image active areas control the bounds of the path-oriented frame.

Method 4: TRANSFER

The output image framing information is transferred from an input reference grid file. This will be the primary method used for framing TIRS images as the product frame will be computed based on the OLI image footprint and then transferred to the TIRS grid. This method would be used if the optional reference grid and reference band parameters are provided as inputs. Note further that this framing method would not be required in the operational implementation if the OLI and TIRS bands are combined in a common grid structure with a common output frame.

The scene framing logic uses the following sub-algorithms/routines:

a) Validate UTM Zone

This routine validates the UTM zone that was entered as an ODL parameter. The scene center longitude will be used for this verification. The nominal UTM zone to use is computed from the scene center longitude but the projection may be forced to an adjacent zone using input parameters. In particular, each WRS path/row may be

preassigned to a UTM zone so that the same zone is always used for scenes near UTM zone boundaries. This should not introduce a zone offset greater than 1. The validation is performed by computing the UTM zone in which the scene center falls and then determining whether the input UTM zone (if any) is within one zone of the nominal zone.

Shift the scene center longitude to put it in the range 0-360 degrees:
SC_long = mod(SC_long + 540, 360)
where: SC_long is the scene center longitude in degrees

Compute the nominal UTM zone (note that UTM zones are six degrees wide):
SC_zone = (int)floor(SC_long/6) + 1

See if the input zone is within one zone of the nominal zone:
if (abs(input_zone - SC_zone) < 2 or (60 - abs(input_zone - SC_zone)) < 2)
then input_zone is valid.

b) North Up Framing
This routine will determine the frame in output space for the north-up product. The actual frame is based on the optimal band's pixel size, but the frame is the same for every band. The method used to determine the scene corners depends on whether the corners were user input (PROJBOX) or calculated by projecting the Level 1R image corners (MINBOX, MAXBOX) but the framing logic is essentially the same in each case. Once input or computed, the latitude/longitude scene corners are converted to the defined map projection, the extreme X and Y coordinates are found, and these extreme points are rounded to a whole multiple of the pixel size. The north-up framing methods are each described in the following sub-algorithms.

b.1) Map Edge/PROJBOX Framing
Calculates the minimum and maximum projection coordinates for given upper-left and lower-right latitude, longitude coordinates.

- Calculate min/max coordinates along east edge of output area by computing latitude/longitude to map x/y projections for a series of points from (minimum latitude, maximum longitude) to (maximum latitude, maximum longitude).
- Calculate min/max coordinates along west edge of output area by computing latitude/longitude to map x/y projections for a series of points from (minimum latitude, minimum longitude) to (maximum latitude, minimum longitude).
- Calculate min/max coordinates along south edge of output area by computing latitude/longitude to map x/y projections for a series of points from (minimum latitude, minimum longitude) to (minimum latitude, maximum longitude).
- Calculate min/max coordinates along north edge of output area by computing latitude/longitude to map x/y projections for a series of points from (maximum latitude, minimum longitude) to (maximum latitude, maximum longitude).

Note that since lines of constant latitude and/or longitude may be curved in map projection space, the extreme map x/y points may not correspond to the four PROJBOX corners.

b.2) Minbox/Maxbox Framing Determine the frame in output space for the minbox or maxbox north-up product. The actual frame is determined based on the optimal band's pixel size, but the frame is the same for every band.

b).2.1 Minbox Framing Calculate the MINBOX frame bounds using the active area corner points for each band.

1. Call projtran (see below) to get the output map projected x/y, for each active area corner point for each image band.
2. Find the minimum and maximum output proj x/y from the full set of active area corner points.
3. Pad the min and max output projection x/y to make them a multiple of pixsize.
4. Fill in the corners for the grid in the order of UL, LL, UR, LR and Y/X coords.
   UL = min x, max y
   UR = max x, max y
   LL = min x, min y
   LR = max x, min y
5. Find the number of lines and samples for the grid, for each specified band number.
   lines = (max y - min y)/pixsize + 1
   samples = (max x - min x)/pixsize + 1

b).2.2 Maxbox Framing Calculate the MAXBOX product frame bounds using the projected corners of each band/SCA.

1. Find the four image corners in input space for each SCA and band.
   UL - (1, first_pixel)
   UR - (1, last_pixel)
   LL - (NLines, first_pixel)
   LR - (NLines, last_pixel)
2. Call the forward model (see below) to get the output lat/long, for each corner point.
3. Call projtran (see below) to get the output map projected x/y, for each corner point.
4. Find the minimum and maximum output proj x/y from the full set of corner points.
5. Pad the min and max output projection x/y to make them a multiple of pixsize.
6. Fill in the corners for the grid in the order of UL, LL, UR, LR and Y/X coords.
   UL = min x, max y
   UR = max x, max y
   LL = min x, min y
   LR = max x, min y

7. Find the number of lines and samples for the grid, for each specified band number.
   lines = (max y - min y)/pixsize + 1
   samples = (max x - min x)/pixsize + 1
8. Call projtran to convert the map projection Y/X coordinates of the output product corners to latitude/longitude.
9. Replace the active area corner coordinates for each band with the converted output product corner coordinates.

b.2.3) Pad Corners Pad the input corners by a defined factor of the pixel size. The x/y min and max values are input for the corner locations. These values are padded by PADVAL * the pixel size. The newly padded x/y min and max values are returned, replacing the original values.

ixmin = int (Xmin/(PADVAL*pixsize))
Xmin = ixmin*PADVAL*pixsize
ixmax = int (Xmax/(PADVAL*pixsize))+1
Xmax = ixmax*PADVAL*pixsize
iymin = int (Ymin/(PADVAL*pixsize))
Ymin = iymin*PADVAL*pixsize
iymax = int (Ymax/(PADVAL*pixsize))+1
Ymax = iymax*PADVAL*pixsize

c) Path-oriented Framing
Provide a path-oriented projection that is framed to a nominal WRS scene. The projection, pixel size, and the path and row of the scene must be defined.

c.1) Calculate Center and Rotation Angle
Calculate the scene center and rotation angle for a nominal WRS scene. The WRS path and row of the input scene and the projection parameters are needed as input. The nominal WRS scene center lat/long and rotation angle for the given projection are returned. The algorithm has the following steps:

Convert input angles to radians:
Inclination_Angle_R = Pi / 180 * Inclination_Angle
Long_Path1_Row60_R = Pi / 180 * Long_Path1_Row60

Compute the Earth's angular rotation rate:
earth_spin_rate = 2 * Pi / (24 * 3600)

Note: We use the solar rotation rate rather than the sidereal rate in order to account for the orbital precession, which is designed to make the orbit sun synchronous. Thus, the apparent Earth angular velocity is the inertial (sidereal) angular velocity plus the precession rate, which, by design, is equal to the solar angular rate.

Compute the spacecraft's angular rotation rate:
SC_Ang_Rate = 2 * Pi * WRS_Cycle_Orbits / (WRS_Cycle_Days*24*3600)

Compute the central travel angle from the descending node:
Central_Angle = (Row - Descending_Node_Row)/Scenes_Per_Orbit*2*Pi

Compute the WRS geocentric latitude:
WRS_GCLat = asin(-sin(Central_Angle) * sin(Inclination_Angle_R))

Compute the longitude of Row 60 for this Path:
Long_Origin = Long_Path1_Row60_R - (Path-1) * 2*Pi/WRS_Cycle_Orbits

Compute the WRS longitude:
Delta_Long = atan2(tan(WRS_GCLat)/tan(Inclination_Angle_R),
cos(Central_Angle)/cos(WRS_GCLat))
WRS_Long = Long_Origin - Delta_Long - Central_Angle *
Earth_Spin_Rate / SC_Ang_Rate

Make sure the longitude is in the range +/- Pi:
While (WRS_Long > Pi)
WRS_Long = WRS_Long - 2*Pi
While (WRS_Long < -Pi)
WRS_Long = WRS_Long + 2*Pi

Compute the scene heading:
Heading_Angle = atan2(cos(Inclination_Angle_R)/cos(WRS_GCLat),
-cos(Delta_Long)*sin(Inclination_Angle_R))

Convert the WRS geocentric latitude to geodetic latitude:
WRS_Lat = atan(tan(WRS_GCLat) * (Semi_Major_Axis/Semi_Minor_Axis) *
(Semi_Major_Axis/Semi_Minor_Axis))

Convert angles to degrees:
WRS_Lat = WRS_Lat * 180 / Pi
WRS_Long = WRS_Long * 180 / Pi
Heading_Angle = Heading_Angle * 180 / Pi

Round WRS lat/long off to the nearest whole arc minute:
WRS_Lat = round(WRS_Lat*60) / 60
WRS_Long = round(WRS_Long*60) / 60

c.2) Calculate Path-oriented Frame
Calculate the center point and rotation angle, and the image corner coordinates in an SOM or UTM projection. Also calculate the first-order polynomial coefficients, which map output line/sample coordinates to their corresponding output projection coordinates. Determine the frame in output space for the path-oriented product. Calculate the frame for each band. The frame must be the same for all bands.

c.2.1) Angle to Map

Convert the WRS rotation angle (from geodetic north) to a frame orientation angle in map coordinates. The following is an algorithm to compute this:

Convert the WRS scene center latitude/longitude to map projection x/y (X1, Y1) using the projtran routine.

Add 1 microradian (0.2 seconds) to the WRS scene center latitude and convert this point to map projection x/y (X2, Y2).

Compute the azimuth of this line in grid space as the arctangent of (X2-X1)/(Y2-Y1). This is the grid azimuth of geodetic north at the WRS scene center.

Add this angle to the WRS rotation angle to give the grid heading. A standard framed scene puts the satellite direction of flight at the bottom of the scene, so the scene orientation angle is the grid heading + or - 180 degrees. If the grid heading is <0 then subtract 180 degrees. If the grid heading is >0 then add 180 degrees. This is the scene orientation angle to use with the WRS scene center.

c.2.2) Path-oriented Minbox/Maxbox Frame

Calculate the path-oriented frame that is large enough to contain all bands.

c).2.2.1. Calculate Path-oriented Minbox Frame

Calculate path-oriented frame for the minbox approach.

1. Compute the map projection coordinates of the four image active area corners for each band as described in step 1 of Minbox Framing.
2. Offset and rotate the scene corners to the path-oriented frame using the WRS scene center map projection coordinates (X1, Y1) and orientation angle:
    a. X' = (X - X1) cos(angle) - (Y - Y1) sin(angle) + X1
    b. Y' = (X - X1) sin(angle) + (Y - Y1) cos(angle) + Y1
3. Compute the minbox frame as described in steps 2-4 of Minbox Framing.
4. Convert the rotated minbox corners back to the unrotated map projection coordinate system:
    a. X = (X' - X1) cos(angle) + (Y' - Y1) sin(angle) + X1
    b. Y = -(X' - X1) sin(angle) + (Y' - Y1) cos(angle) + Y1

c.2.2.2) Calculate Path-oriented Maxbox Frame

Calculate path-oriented frame for the maxbox approach.

1. Compute the map projection coordinates of the four image corners as described in steps 1-3 of Maxbox Framing.
2. Offset and rotate the scene corners to the path-oriented frame using the WRS scene center map projection coordinates (X1, Y1) and orientation angle:
    a. X' = (X - X1) cos(angle) - (Y - Y1) sin(angle) + X1
    b. Y' = (X - X1) sin(angle) + (Y - Y1) cos(angle) + Y1
3. Compute the maxbox frame as described in steps 4-7 of Maxbox Framing.

4. Convert the rotated maxbox corners back to the unrotated map projection coordinate system:
    a. X = (X' - X1) cos(angle) + (Y' - Y1) sin(angle) + X1
    b. Y = -(X' - X1) sin(angle) + (Y' - Y1) cos(angle) + Y1
5. Call projtran to convert the map projection Y/X coordinates of the output product corners to latitude/longitude.
6. Replace the active area corner coordinates for each band with the converted output product corner coordinates.

d) Transfer Framing
Open and read the framing information from an input reference grid file. The map projection, image size, image bounds, pixel size, and other framing parameters are copied from the specified band in the reference grid instead of being computed using one of the other methods described above. This method will be used to transfer OLI scene frames to the corresponding TIRS data.

e) Celestial Acquisitions
Celestial acquisitions use the same framing logic as Earth acquisitions (namely maxbox) but the output space coordinate systems are sufficiently different to merit separate discussion. For both lunar and stellar acquisitions the output space is defined in terms of directions in inertial space, defined by the ECI J2000 right ascension and declination of the TIRS look vectors. In the case of stellar acquisitions, the output space "projection" uses the ECI J2000 right ascension and declination directly. For lunar acquisitions the output coordinate system is modified to use the LOS right ascension and declination offset from the lunar right ascension and declination at the time of observation. This creates a slowly rotating coordinate system that tracks the moon and is the reason for having a planetary ephemeris file as an input to this algorithm. These differences emerge in the forward model computations for celestial acquisitions where the LOS intersection logic used for Earth acquisitions is replaced by operations on the inertial lines-of-sight (after conversion to inertial right ascension and declination angles), with the resulting map projection x/y coordinates used in the Earth-view algorithms replaced by right ascension and declination (or delta-right ascension and delta-declination). The same maxbox framing logic applied to the x/y map projection coordinates in Earth-view acquisitions is then applied to these angular celestial coordinates.

4.3.2.6.3 Stage 3 - Grid Definition
The grid definition stage determines the required size of the grid, allocates the grid structure, and computes the input space (Level 1R) line/sample locations for each grid cell.

a) Determine Number of Grid Input/Output Lines/Samples
This routine will determine the number of input points to be stored in the grid according to the grid sampling rate or grid cell size chosen.

Loop through each band stored in the grid

Loop through each SCA stored in the grid.

Calculate the number of lines and samples stored in the grid according to the size of each grid cell and the size of the input image to be processed. Store the number of grid lines and samples calculated in the grid.

Calculate number of times grid cell size divides into Level 1R imagery

$$\text{number of grid lines} = \frac{\text{number of image lines}}{\text{grid cell size line direction}} + 1$$

$$\text{number grid samples} = \frac{\text{number of detectors per SCA}}{\text{grid cell size sample direction}} + 1$$

where:

number of image lines = number of lines in Level 1R (LOS model)
number of detectors per SCA = number of samples per SCA (LOS model)
grid cell size line direction = number of lines in one grid cell
grid cell size sample direction = number of samples in one grid cell

If the grid cell size in the line direction does not divide evenly into the number of lines in the Level 1R then increment the number of grid lines by one.

If the grid cell size in the sample direction does not divide evenly into the number of samples in the Level 1R then increment the number of grid samples by one.

b) Determine Grid Lines/Samples

Given the number of grid lines and samples that will be sampled in the input imagery, this routine calculates where each grid cell point will fall in the input Level 1R image. These grid cell points will fall at integer locations in the input imagery.

Loop through each band that is stored in the grid

Loop through each SCA stored in the grid

Initialize first grid cell line location to zero relative.

input line location grid $cell_0$ = 0

Loop until the grid cell line location is greater than or equal to the number of Level 1R lines, incrementing each new grid cell line location by the appropriate grid cell size in the line direction for the current band and SCA.

input line location grid $cell_n$ = input line location grid $cell_{n-1}$ + grid cell size line direction

Set last grid cell line location to the last line in Level 1R image.

$$\text{input line location grid cell}_{last} = \text{number of lines in Level 1R imagery}$$

Initialize first grid cell sample location to zero relative.

$$\text{input sample location grid cell}_0 = 0$$

Loop until the grid cell sample location is greater than or equal to the number of Level 1R samples, incrementing each new grid cell sample location by the appropriate grid cell size in the sample direction for the current band and SCA.

$$\text{input sample location grid cell}_n = \text{input sample location grid cell}_{n-1} + \text{grid cell size sample direction}$$

Set last grid cell sample location to the last sample in Level 1R image.

$$\text{input sample location grid cell}_{last} = \text{number of samples in Level 1R imagery}$$

4.3.2.6.4 Stage 4 - Grid Construction

Once the grid structures are created (one per SCA per band) the forward model is evaluated at every grid intersection, that is, for every Level 1R line/sample location at every elevation plane. The forward model computes the WGS84 latitude/longitude coordinates associated with each input line/sample/height point. These latitude/longitude positions are then converted to output space line/sample by projecting them to map x/y, computing the offsets (and rotation if path-oriented) from the upper-left scene corner, and scaling the offsets from meters to pixels using the pixel size.

a) Make Grid

This routine creates the geometric mapping grid in output space.

Given the number of grid lines and samples that will be sampled in the input imagery, loop on each band of each SCA, loop on number of z-planes, loop on number of input grid lines and samples calculating the corresponding output line and sample location. For each input line, sample location, and elevation, the instrument forward model function is called. This forward model function is outlined in the steps below. Additional detail on the sub-algorithms that comprise the forward model is provided in the subsection titled "Forward Model" later in this document.

The forward model uses the TIRS LOS model structure and the CPF to map an input line and sample location to an output geographic location. These are the steps that are performed whenever calculating an output geodetic latitude and longitude from an input line and sample by invoking the instrument “forward model.” The GCTP function can then be used to transform the geographic latitude and longitude to a map projection X and Y coordinate. If the output image has a “North up” orientation, then the upper-left projection coordinate of the output imagery and the output pixel size can be used to transform any projection coordinate to an output line and sample location. If the map

projection space is in a rotated projection space, such as having a satellite path orientation, then a transformation handling rotation is established between projection space and output pixel location. This transformation is then used in converting projection coordinates to output pixel line and sample locations.

The process listed below is performed on all bands, all elevation planes, and all SCAs present in the grid. The detector type used in the process is nominal (see the TIRS LOS Model Creation Algorithm (4.3.1) for a discussion of detector types). The list explains the actions taken if a detector type other than nominal is chosen, so that it can be referenced later.

Loop on number of input grid lines.

Loop on number of input grid samples.

Read the input space (Level 1R) line/sample coordinate for this grid point.

Loop on the number of elevation planes.

Compute the height of the current elevation plane:

$$height = (z - z_{elev=0}) * elevation\_increment$$

where:
z is the index of the current z-plane and
$z_{elev=0}$ is the index of the zero elevation z-plane.

Invoke the forward model to compute the corresponding ground position latitude/longitude for this point. The general steps of the forward model are described here and are presented in more detail below.

Find Time
Find the nominal time of input sample relative to the start of the imagery. This procedure is described in the TIRS LOS Model Creation Algorithm (4.3.1) and is repeated below in the Find Time sub-algorithm description.

Find TIRS LOS
Find the TIRS LOS vector for the input line/sample location using the Legendre polynomial coefficients and the scene select mirror angle as described below in the Find TIRS LOS sub-algorithm.

Find Attitude
Calculate the spacecraft attitude corresponding to the LOS, i.e., for the line/sample location, at the time computed above, using the Find Attitude sub-algorithm described below. Note that for Earth acquisitions the roll-pitch-yaw attitude sequence in the LOS model is relative to the orbital coordinate

system whereas for celestial (lunar/stellar) acquisitions the LOS model roll-pitch-yaw sequence is with respect to the ECI J2000 coordinate system. The operations applied by the Find Attitude sub-algorithm are the same in either case.

Find Ephemeris
Calculate satellite position for line/sample using Lagrange interpolation. Reference the move_sat sub-algorithm described in the TIRS LOS Model Creation Algorithm (4.3.1) and repeated below. Note that for Earth acquisitions the move_sat sub-algorithm is provided with the corrected ECEF ephemeris data from the LOS model whereas for celestial (lunar/stellar) acquisitions it will be passed the corrected ECI ephemeris.

Rotate LOS to ECEF (Earth-view) or ECI (Celestial)
Use the TIRS alignment matrix in the TIRS LOS model to convert the LOS vector from sensor to ACS/body coordinates. Then apply the interpolated roll, pitch, and yaw to the LOS to convert ACS/body to orbital (Earth-view) or ECI (celestial). If Earth-view, use the ephemeris to construct the orbital to ECEF rotation matrix and use it to transform LOS to ECEF. The procedure for Earth-view scenes is described in the Attitude sub-algorithm below. For celestial acquisitions, the procedure is complete once the LOS has been rotated to ECI using the roll-pitch-yaw perturbation matrix.

Spacecraft Center of Mass to TIRS Offset Correction
Adjust the spacecraft position for the offset between the spacecraft center of mass and the TIRS instrument. This offset, in spacecraft body coordinates, is stored in the LOS model structure. First, convert the offset from spacecraft body frame to ECEF using the attitude perturbation matrix (body to orbital) and the orbital to ECEF matrix:

[orbital CM to TIRS] = [perturbation] [body CM to TIRS]

[ECEF CM to TIRS] = [ORB2ECEF] [orbital CM to TIRS]

Add the offset to the ECEF spacecraft position vector. This correction is not used for celestial (lunar/stellar) acquisitions.

Correct LOS for Velocity Aberration
The relativistic velocity aberration correction adjusts the computed LOS (ECEF for Earth-view and ECI for celestial) for the apparent deflection caused by the relative velocity of the platform (spacecraft) and target. The preparatory computations are somewhat different for Earth-view and celestial acquisitions due to the differences in target velocity.

Earth-view Case
The LOS intersection sub-algorithm described below (see Find Target Position) is invoked with an elevation of zero to find the approximate ground target position. The ground point velocity is then computed as:

$\mathbf{V}_g = \omega \times \mathbf{X}_g$

where:

$\mathbf{V}_g$ = ground point velocity
$\mathbf{X}_g$ = ground point ECEF position
ω = Earth rotation vector = $[\ 0 \quad 0 \quad \Omega_e\ ]^T$
Ω_e = Earth rotation rate in radians/second (from CPF)

The relative velocity is then:

$\mathbf{V} = \mathbf{V}_s - \mathbf{V}_g$

where Vs is the spacecraft ECEF velocity from the ephemeris data.

Correcting the Earth-View LOS
The LOS vector is adjusted based on the ratio of the relative velocity vector to the speed of light (from the CPF):

$$\boldsymbol{l}' = \frac{\boldsymbol{l} - \frac{\boldsymbol{V}}{c}}{\left| \boldsymbol{l} - \frac{\boldsymbol{V}}{c} \right|}$$

where: **l** = uncorrected LOS and **l'** = corrected LOS

Note that in this case the LOS velocity aberration correction is negative since we are correcting the apparent LOS to the true (aberration corrected) LOS. The correction is positive if we are computing the apparent LOS from the true (geometrical) LOS (see lunar case below).

Celestial (Lunar/Stellar) Case
Both lunar and stellar acquisitions use the spacecraft inertial velocity from the ephemeris data as the relative velocity. This is justified by the use of a lunar ephemeris (using the Naval Observatory's NOVAS-C package) that returns apparent places. The apparent location of the moon is already corrected for light travel time (see below) and velocity/planetary aberration due to the motion of the moon around the Earth. Thus, the residual aberration is due only to the motion of the spacecraft relative to the Earth. Thus, for both lunar and stellar acquisitions:

$\mathbf{V} = \mathbf{V}_s$

where Vs is the spacecraft ECI velocity from the ephemeris data.

Correcting the Celestial LOS

For stellar acquisitions, the LOS is corrected for aberration in the same manner as for Earth-view scenes. For lunar acquisitions, rather than correct the LOS vector, we adjust the apparent location of the moon. The lunar vector is thus adjusted based on the ratio of the relative velocity vector to the speed of light (from the CPF) as:

$$\mathbf{l}' = \frac{\mathbf{l} + \frac{\mathbf{V}}{c}}{\left| \mathbf{l} + \frac{\mathbf{V}}{c} \right|}$$

where: **l** = uncorrected LOS and **l'** = corrected LOS

The correction is positive in this case since we are computing an apparent location rather than correcting one.

LOS Intersection

For Earth-view acquisitions, intersect the LOS in ECEF with the Earth model as described in the Find Target Position sub-algorithm below. This yields the geodetic latitude, longitude, and height of the ground point.

For celestial acquisitions, convert the ECI LOS to right ascension (RA) and declination (δ) angles:

$$RA = \tan^{-1}\left(\frac{y}{x}\right)$$

$$\delta = \tan^{-1}\left(\frac{z}{\sqrt{x^2 + y^2}}\right)$$

where the ECI los vector is $[\, x \;\; y \;\; z \,]^T$.

Correct Ground Position for Light Travel Time

Since the light departing the ground point takes a finite time to arrive at the TIRS sensor, there is a slight discrepancy in the corresponding time at the ground point and at the spacecraft. Since the LOS intersection logic assumed that these times were the same, a small correction can be made to correct for this light travel time delay.

Given the ECEF positions of the ground point and the spacecraft, compute the light travel time correction as follows:

Compute the distance from the ground point to the spacecraft:

$$d = \left| \mathbf{X}_s - \mathbf{X}_g \right|$$

where:

$\mathbf{X}_s$ is the spacecraft ECEF position and

$\mathbf{X}_g$ is the ground point ECEF position.

Compute the light travel time using the speed of light (from CPF):

$$ltt = \frac{d}{c}$$

Compute the Earth rotation during light travel:

$\theta = ltt * \Omega_e$ where Ω_e is the Earth angular velocity from the CPF.

Apply the light travel time Earth rotation:

$$\mathbf{X}'_g = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{X}_g$$

where:

$\mathbf{X}_g$**'** is the corrected ECEF position
X_g is the uncorrected ECEF position

Convert the corrected ECEF position to geodetic latitude, longitude and height.

Note that the light travel time correction for lunar observations due to the difference between the Earth-moon distance and the spacecraft-moon distance is neglected. This is justified by the fact that that the lunar angular rate is less than 3 microradians per second and the maximum LTT difference is about 25 milliseconds making the magnitude of this effect less than 0.1 microradians.

Convert Position to Output Space Line/Sample

The angular geodetic (latitude/longitude) or celestial (RA/declination) coordinates must be converted to the corresponding output space line/sample coordinate to complete the input space to output space mapping.

For Earth-view acquisitions this is accomplished as follows:

Calculate the map projection X/Y for the geodetic latitude and longitude.

Convert map X/Y coordinate to output line/sample location:

If the output map projection is of a path-oriented projection then the X/Y coordinate is transformed to output space with a bilinear transformation.

$$line = a_0 + a_1 * X + a_2 * Y + a_3 * X * Y$$
$$sample = b_0 + b_1 * X + b_2 * Y + b_3 * X * Y$$

where:
a_i = polynomial coefficients that map X/Y to an output line location
b_i = polynomial coefficients that map X/Y to an output sample location
X,Y = map projection coordinates

The polynomial transformation is set up to handle the rotation involved in rotating a “Map North” projection to Satellite of “Path” projection (i.e., one that has the output line coordinate system more closely aligned with the along flight path of the satellite).

If the output map projection is not path-oriented, but “North up,” the relationship between X/Y and output line/sample does not involve any rotation and the following equation is used:

$$\text{line} = \frac{\text{upper left Y} - \text{Y}}{\text{pixel size Y}}$$

$$\text{sample} = \frac{\text{X} - \text{upper left X}}{\text{pixel size X}}$$

where:

upper left Y = upper-left Y projection coordinate of output image
upper left X = upper-left X projection coordinate of output image
pixel size Y = output pixel size in Y coordinates
pixel size X = output pixel size in X coordinates

Note that these line and sample pixel coordinates are (0,0) relative (i.e., the center of the upper-left pixel is at line,sample 0,0).

For lunar acquisitions, the right ascension and declination angles derived from the inertial LOS are offset from the nominal lunar inertial position to establish an output frame that "tracks" the apparent location of the moon. This is done as follows:

a) Compute the apparent ECI J2000 position of the moon.
1. Use the input JPL lunar ephemeris data in the NOVAS-C package to compute the ECITOD apparent location of the moon at the time corresponding to the current LOS (lxx_moonpos). This apparent location is provided as an ECITOD vector (i.e., including both direction and distance).
2. Apply the nutation and precession corrections (see Ancillary Data Preprocessing Algorithm (4.1.4) for additional information) to convert the ECITOD vector to ECI J2000.
3. Subtract the current spacecraft ECI J2000 position vector from the lunar ECI J2000 vector to compute the spacecraft-lunar vector.

4. Compute the apparent (parallax corrected) right ascension, declination, and spacecraft-lunar distance from the spacecraft-lunar vector (by invoking exx_cart2sph.

b) Compute the differences between the LOS right ascension and declination and the apparent lunar right ascension and declination.

c) Normalize the nominal angular pixel size by the ratio of the current spacecraft-moon distance (computed above) and the nominal spacecraft-moon distance. The nominal distance is computed at the acquisition center time.

$$psize_{current} = psize_{nominal} * distance_{nominal} / distance_{current}$$

d) Divide the angular distances computed in b) above by the normalized pixel size computed in c) above. This yields the moon-relative line/sample coordinate. This is the coordinate space in which lunar images are framed, so the offset between these coordinates and the lunar scene upper-left corner coordinates yields the output space line/sample for the current grid point.

For stellar acquisitions, the right ascension and declination angles derived from the inertial LOS are used directly. The offsets relative to the scene upper-left corner (in right ascension/declination space) are computed and divided by the angular pixel size to compute output space line/sample coordinates.

One additional note regarding the celestial acquisition scene framing is in order. Since right ascension, like longitude, increases eastward, and declination, like latitude, increases northward, and given that celestial images are looking "up" rather than "down," the right ascension-x, declination-y coordinate system is left-handed. This can lead to the moon being apparently inverted left-to-right in the output image. This is not important for the applications (e.g., band registration characterization) in which the lunar images are to be used. If "anatomically correct" lunar images are required, some changes to the framing logic may be necessary.

The line and sample location calculated is stored in the grid structure. This line/sample location is then the output location for the corresponding input line/sample and the current elevation (current grid line/sample input locations).

b) Calculate Jitter Sensitivity Coefficients

The forward model is invoked multiple times at each grid intersection to compute the effect that small attitude perturbations about each spacecraft axis have on the input space to output space line/sample mapping. This is done at each grid point as follows:

Save the current grid point input line/sample as in_line/in_samp and the current grid point output line/sample as line0/samp0.

For each spacecraft axis (roll-pitch-yaw):

1. Perturb the attitude about the selected axis by 1 microradian.
2. Use the forward model to compute the output line/sample corresponding to the current input line/sample using the perturbed attitude and store the result in line[0]/samp[0].
3. Perturb the input line number by 1 line (delta_line = 1) and recompute the corresponding output line/sample, storing the result in line[1]/samp[1].
4. Restore the input line number to in_line and perturb the input sample number by 1 sample (delta_samp = 1) and recompute the corresponding output line/sample, storing the result in line[2]/samp[2].
5. Calculate the output space to input space line/sample sensitivities as:
   a. delta_oline_per_iline = (line[1]–line[0]) / delta_line
   b. delta_oline_per_isamp = (line[2]–line[0]) / delta_samp
   c. delta_osamp_per_iline = (samp[1]–samp[0]) / delta_line
   d. delta_osamp_per_isamp = (samp[2]–samp[0]) / delta_samp
6. Invert the resulting 2-by-2 sensitivity matrix to find the input line/samp per output line/samp sensitivities:
   a. determinant = delta_oline_per_iline * delta_osamp_per_isamp – delta_oline_per_isamp * delta_osamp_per_iline
   b. delta_iline_per_oline = delta_osamp_per_isamp / determinant
   c. delta_iline_per_osamp = -delta_oline_per_isamp / determinant
   d. delta_isamp_per_oline = -delta_osamp_per_iline / determinant
   e. delta_isamp_per_osamp = delta_oline_per_iline / determinant
7. Apply the input line/samp per output line/samp sensitivities to the output line/samp offset due to the attitude perturbation, to find the equivalent input space offset :
   a. d_iline = delta_iline_per_oline * (line[0] – line0) + delta_iline_per_osamp * (samp[0] – samp0)
   b. d_isamp = delta_isamp_per_oline * (line[0] – line0) + delta_isamp_per_osamp * (samp[0] – samp0)
8. Divide by the attitude perturbation to compute the input line/sample to attitude jitter sensitivities for this axis at this grid point:
   a. line_sens[axis] = -d_iline / perturbation
   b. samp_sens[axis] = -d_isamp / perturbation

   Where :

   line_sens[] is the array of roll-pitch-yaw line sensitivities for the grid.
   samp_sens[] is the array of roll-pitch-yaw sample sensitivities for the grid.
   perturbation is the 1 microradian attitude perturbation introduced in step 1.

   Note that the sign of the sensitivities is inverted in this calculation. This is done because the sensitivities will be used to compute the equivalent input space corrections needed to compensate for an attitude disturbance. Therefore, since d_iline is the input space line offset that is equivalent to one microradian of jitter for the current axis, an offset of –d_iline will compensate for this jitter.

A 2-by-3 array containing the line and sample sensitivity coefficients for the roll, pitch, and yaw axes is stored for each grid point.

c) Calculate Map Coefficients

Bilinear mapping coefficients for each grid cell are calculated for mapping from input location to output location (forward mapping) and for mapping from output location to input location (inverse mapping). A separate mapping function is used for lines and samples. This equates to four mapping functions. A set of four mapping functions is calculated for each grid cell, for each SCA, for every band, and for every elevation plane that is stored in the grid.

The following methodology is used for calculating one set of four bilinear mapping equations:

A 9x4 matrix is used to fit nine points within a grid cell. The matrix equation takes the form of:

$$[A][coeff\ ] = [b]$$

In this equation, matrix A is 9x4, vector b is 9x1, and the coefficient matrix is 4x1. The coefficient matrix, [*coeff*], can be solved to obtain the mapping coefficients as:

$$[coeff] = [A^T A]^{-1}[A^T b]$$

In the case of solving for an equation to map an input line and sample location to an output sample location, belonging to one grid cell, the matrices can be defined as:

$A_{n,0} = 1$ where n=0,8

$A_{0,1}$ = upper-left input sample location for current grid cell
$A_{1,1}$ = upper-right input sample location for current grid cell
$A_{2,1}$ = lower-left input sample location for current grid cell
$A_{3,1}$ = lower-right input sample location for current grid cell
$A_{4,1} = (A_{0,1}+A_{1,1}+A_{2,1}+A_{3,1})/4$
$A_{5,1} = (A_{0,1}+A_{1,1})/2$
$A_{6,1} = (A_{1,1}+A_{3,1})/2$
$A_{7,1} = (A_{2,1}+A_{3,1})/2$
$A_{8,1} = (A_{2,1}+A_{0,1})/2$
$A_{0,2}$ = upper-left input line location for current grid cell
$A_{1,2}$ = upper-right input line location for current grid cell
$A_{2,2}$ = lower-left input line location for current grid cell
$A_{3,2}$ = lower-right input line location for current grid cell
$A_{4,2} = (A_{0,2}+A_{1,2}+A_{2,2}+A_{3,2})/4$
$A_{5,2} = (A_{0,2}+A_{1,2})/2$
$A_{6,2} = (A_{1,2}+A_{3,2})/2$
$A_{7,2} = (A_{2,2}+A_{3,2})/2$
$A_{8,2} = (A_{2,2}+A_{0,2})/2$
$A_{n,3} = A_{n,1} * A_{n,2}$ where n=0…8

b_0 = upper-left output sample location for current grid cell

b_1 = upper-right output sample location for current grid cell
b_2 = lower-left output sample location for current grid cell
b_3 = lower-right output sample location for current grid cell
$b_4 = (b_0+b_1+b_2+b_3)/4$
$b_5 = (b_0+b_1)/2$
$b_6 = (b_1+b_3)/2$
$b_7 = (b_2+b_3)/2$
$b_8 = (b_2+b_0)/2$

The line and sample locations listed above are defined at the grid cell corners coordinates. The points interpolated in between the grid cell line segments provide stability for what could be, most notably a mapping that involves a 45° rotation, an ill-defined solution if only four points were used in the calculation. The set of coefficients define a bilinear mapping equation of the form:

$$sample_o = coeff_0 + coeff_1 * sample_i + coeff_2 * line_i + coeff_3 * sample_i * line_i$$

where:
$sample_o$ = output sample location
$sample_i$ = input sample location
$line_i$ = input line location

The forward mapping equations, mapping input line and sample locations to output line locations can be solved by swapping output line locations for output sample locations in the matrix [b]. The reverse mapping equations, mapping output locations to input line and sample, can be found by using output line and sample locations in the [A] matrix and the corresponding input sample and then line locations in the [b] matrix.

c.1) Calculate Forward Mappings
This function, given grid points in both input and output space, uses the Calculate Map Coefficients algorithm described above to generate the mapping polynomial coefficients needed to convert from a line/sample in input space (satellite) to one in output space (projection). It generates these coefficients for every cell in the grid.

c.2) Calculate Inverse Mappings
This function, given grid points in both input and output space, uses the Calculate Map Coefficients algorithm described above to generate the mapping polynomial coefficients needed to convert from a line/sample in output space (projection) to one in input space (satellite). It generates these coefficients for every cell in the grid.

4.3.2.6.5 Stage 5 - Finalize the Grid

The final stage of grid processing generates the global (rough) mapping coefficients, used to initially identify the appropriate grid cell, and computes the parallax sensitivity coefficients, used to correct for even/odd detector offset effects, for each grid cell.

a) Calculate Rough Mapping Coefficients
This routine will find the rough mapping coefficients for the grid. The rough polynomial is a set of polynomials used to map output line and sample locations to input line and sample locations. The rough polynomial is generated using a large number of points distributed over the entire scene, and by calculating a polynomial equation that maps an output location to an input location. The rough polynomial is only meant to get a “close” approximation to the input line and sample location for a corresponding output line and sample location. Once this approximation is made, the value can be refined to get a more accurate solution. A rough mapping polynomial is found for every SCA, for every band, and for every elevation plane that is stored in the grid file.

Bilinear mapping was found to be sufficient for the rough mapping. The mapping function therefore looks like the ones used for each individual grid cell. However, the setup of the matrices to solve for the mapping coefficients is different:

$$\underset{Nx4}{[A]}\underset{4x1}{[coeff]}=\underset{Nx1}{[b]}$$

Where the matrix [A] is defined by the output line and sample locations, matrix [b] is defined by either the input lines or input samples, and N is equal to the total number of points stored in the grid for one elevation plane, of one band, for a single SCA. The rough polynomial is therefore found by using all the point locations stored in the grid for a given band and elevation plane for a single SCA. There is one mapping for output line and sample location to input sample location and one mapping for output line and sample location to input line location.

Grid Cell Polynomial
This utility function calculates a "rough" mapping of output to input lines/samples. The coefficients returned from this function are used as a first order approximation to an inverse line-of-sight model. This polynomial is used to initially locate the grid cell to be used in the resampling process, providing a starting point for the more accurate inverse model based on individual grid cell parameters.

b) Calculate Detector Offsets
This function computes the detector offset values and stores linear mapping coefficients associated with detector offsets in the grid structure. Using the zero elevation plane, for each band and each SCA, loop on the input lines and samples calculating the detector offsets. The detector offsets are set up to account for the geometric differences between the primary and redundant rows of detectors and the “nominal” set of detectors modeled by the Legendre polynomials (see the TIRS LOS Model Creation Algorithm (4.3.1)). These differences are considered to be consistent between actual and nominal detectors when they occur under the same acquisition conditions, i.e., they are slowly varying. These actual to nominal detector differences are due to the imperfect trade-off between space (detector offset) and time (detector delay) that is made when we temporally shift (through the use of Level 1R image fill) the deselected/replaced detectors to compensate for their spatial offsets on the focal plane. The degree to which

this time/space trade is imperfect varies with height and, so, the corrections derived here and stored in the grid structure, are functions of detector offset and height.

There are also the subpixel detector-specific offsets that are stored in the CPF. These "exact" detector-specific offsets are accounted for in the resampling process. Note that the potential for deselected detectors has made it necessary to also store per-detector full-pixel offsets in the CPF (and LOS model). As a result, this detector offset sensitivity logic computes the offset sensitivity per pixel of detector offset rather than a fixed value. The routine ols2ils listed below, used for mapping an output line and sample to an input line and sample using the geometric grid, is discussed in the TIRS Image Resampling Algorithm.

Loop on number of bands stored in grid
    Loop on number of SCAs stored in grid
        Loop on lines and samples stored in the grid

            Get the maximum detector offset value for this band from the CPF.

            Calculate the output line/sample location for the current input line and sample and the zero elevation plane, calculated using the forward model (see below) with the detector location set to MAXIMUM. This new detector type is the same as ACTUAL but uses the maximum detector offset rather than the detector-specific value.

            Map calculated output line/sample back to input space using the TIRS LOS grid and ols2ils.

            Delta line/sample per pixel of offset are calculated by:

$$\Delta line_0 = (\text{nominal line} - \text{mapped line}) / \text{max offset}$$
$$\Delta sample_0 = (\text{nominal sample} - \text{mapped sample}) / \text{max offset}$$

            where:
                nominal line = current grid cell line location
                mapped line = input line location from ols2ils mapped "maximum" output line
                nominal sample = current grid cell sample location
                mapped sample = input sample location from ols2ils mapped “maximum” output sample
                max offset = detector offset used in the MAXIMUM forward model calculations

            These delta lines and samples represent the input space correction necessary to compensate for the difference between nominal and

actual detectors per pixel of detector offset, for the zero elevation plane.

Repeat these calculations for the maximum elevation plane to compute $\Delta line_H$ and $\Delta sample_H$ where H is the elevation corresponding to the maximum z-plane.

Compute the line and sample even/odd offset sensitivity coefficients:

$$c_0 = \Delta line_0$$
$$c_1 = (\Delta line_H - \Delta line_0) / H$$
$$d_0 = \Delta sample_0$$
$$d_1 = (\Delta sample_H - \Delta sample_0) / H$$

Note that c_0 and d_0 are in units of pixels per pixel and c_1 and d_1 are in units of pixels per meter per pixel.

These c_i and d_i coefficients are stored in the projection grid to be used during the resampling process.

Output Line/Sample to Input Line/Sample
This utility routine maps an output space line/sample back into its corresponding input space line/sample. This is done using the "rough" polynomial from the grid to determine an initial guess at an input space line and sample. From this initial guess a grid cell row and column is calculated and the inverse coefficients for that cell are retrieved from the grid. These coefficients are used to determine an exact input space line and sample (in extended space).

Find Grid Cell
This utility function finds the correct cell that contains the output line/sample. It finds the correct grid cell containing the output pixel by first determining the set of grid cells to be checked. It then calls a routine to perform a "point in polygon" test on each of these grid cells to determine if the pixel does indeed fall within that grid cell.

Forward Model
Having described the grid generation procedure we now turn to the forward model, referred to extensively above, in more detail.

For a given line, sample and band, propagate the forward model to determine a latitude and longitude for the specified point. This involves finding the time of the observation, constructing the instrument line-of-sight, calculating the spacecraft attitude and ephemeris for the observation time, and intersecting the projected line-of-sight with the Earth’s surface. The entire forward model procedure is referred to as LOS projection and is described systematically below.

a) Project TIRS LOS
This function finds the position where the line of sight vector intersects the Earth's surface. It invokes the following sub-algorithms:

a.1) Find Time
This function finds the time into the scene given the line, sample, and band. The input sample number is 0-relative and relative to the SCA. The accounting for the primary/redundant detector offsets is based on the value of the dettype variable, which may be NOMINAL, ACTUAL, MAXIMUM or EXACT. Note that the EXACT selection is treated the same as ACTUAL. This is because even though fractional-pixel detector offsets can occur, the compensating time shifts implemented by inserting fill pixels can only be introduced in whole-line increments. Therefore, the subpixel difference between the ACTUAL and EXACT detector types affects only the LOS angle not the time. The MAXIMUM detector type represents a theoretical offset that is used to calculate the parallax coefficients within the grid, called MAX_DET_DELAY.

Due to the multiple detector rows and the potential for bad detector replacement, a nominal and an actual time can be found in a scene. If the current position within the image is given as a line and sample location, the two different “types” of times for TIRS pixels are calculated by:

```
if detector type is set to MAXIMUM
    detector_shift_x = maximum_detector_shift
    l0r_fill_pixels = round(detector_shift_x) + nominal_fill
else
    detector_shift_x = shift stored in geometric model
     l0r_fill_pixels = Fill from L0Rp (also stored in geometric model)

time_index = line_number - l0r_fill_pixels
if ( time_index < 0 ) time_index = 0
if (time_index > (num_time_stamps - 1)) time_index = num_time_stamps - 1

actual_time = line_time_stamp[time_index] - integration_time/2
                    + (line_number - l0r_fill_pixels - time_index) * TIRS_sample_time

nominal_time = actual_time + (l0r_fill_pixels – nominal_fill) * TIRS_sample_time
```

where:

- line_number is the zero-referenced TIRS image line number (N).
- nominal_fill is the amount of detector alignment fill to be inserted at the beginning of pixel columns that correspond to nominal detectors; that is, those detectors with a delay value of zero that are the basis for the Legendre polynomial LOS model. This value comes from the CPF and will be zero if there are no bad detectors to replace.
- l0r_fill_pixels is the total amount of detector alignment fill to be inserted at the beginning of the pixel column associated with the current detector. It includes

both the nominal_fill_pixels and the detector-specific delay fill required to align deselected/replaced detectors.

- num_time_stamps is the total number of time codes (image lines) in the image. It is tested to ensure that time_index, the line_time_stamp index, does not go out of bounds.
- detector_shift_x is the amount of detector offset for the current detector from the TIRS LOS model detector delay table. It is rounded to the nearest integer pixel because time offsets can only occur in whole line increments.

The detector_shift_x offset parameter from the LOS model detector delay table is rounded to include the effects of detector deselect/replacement but not the detector-specific subpixel offsets.

a.2) Find TIRS LOS

This function finds the line of sight vector in sensor coordinates, using the Legendre polynomial LOS model and the SSM model stored in the TIRS LOS model, as follows:

Find normalized detector for Legendre polynomial:

$$\text{normalized detector} = \frac{2 * (\text{current detector})}{(\text{number of detectors-1})} - 1$$

where:

current detector = sample location (in the range 0 to number of detectors-1)

number of detectors = number of detectors (samples) for current band and SCA (from TIRS LOS model)

Find across-track (y) and along-track (x) angles:

$$x = coef\_x_0 + coef\_x_1 * (\text{nd}) + coef\_x_2 * (1.5 * (\text{nd})^2 - 0.5) + coef\_x_3 * (\text{nd}) * (2.5 * (\text{nd})^2 - 1.5)$$

$$y = coef\_y_0 + coef\_y_1 * (\text{nd}) + coef\_y_2 * (1.5 * (\text{nd})^2 - 0.5) + coef\_y_3 * (\text{nd}) * (2.5 * (\text{nd})^2 - 1.5)$$

where:

nd = normalized detector

coef_x = Legendre coefficients for along-track direction

coef_y = Legendre coefficients for across-track direction

(Note: coef_x and coef_y are read from the CPF and stored in the LOS model)

If LOS requested is ACTUAL, add the whole pixel detector shift (detector, band, and SCA dependent) from the TIRS LOS model. This detector shift is only in the along-track direction. Note that the TIRS LOS model contains the combined whole pixel and subpixel detector offset, so it must be rounded to the integer part for the ACTUAL detector type and left unrounded for the EXACT detector type.

x = x + round(detector_shift_x) * IFOV

If LOS requested is EXACT, then add individual detector offsets (detector number, band, and SCA dependent). This detector shift is in both the along and across-track directions. These values are stored within the TIRS LOS model.

x = x + (detector_shift_x) * IFOV
y = y + (detector_shift_y) * IFOV

Note that the detector_shift_y parameter, from the TIRS LOS model detector delay table, is always subpixel. See TIRS LOS Model Creation Algorithm (4.3.1) for further explanation of NOMINAL/ACTUAL/EXACT line of sight.

If the LOS request is MAXIMUM then add the maximum detector offset.

x = x + (maximum_detector_shift_x) * IFOV

Calculate LOS vector.

$$[\mathrm{los}] = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Normalize LOS.

$$\vec{\mathrm{los}} = \frac{\vec{\mathrm{los}}}{\left\|\vec{\mathrm{los}}\right\|}$$

Apply the TIRS telescope alignment matrix (from the TIRS LOS model) to the LOS vector.

$$\mathbf{[los}_{SSM}\mathbf{]} = \mathbf{M}_{Tele2SSM}\ \mathbf{[los]}$$

Calculate the SSM angle (from the TIRS LOS model) using the sample time returned from the find time sub-algorithm (nominal or actual). Note that both the image and SSM times are referenced to the same UTC epoch (see TIRS LOS Model Creation Algorithm (4.3.1) for details).

1. In the TIRS SSM model, find the last SSM angle sample, θ_n, with a time, t_n, earlier than the pixel time.
2. Linearly interpolate the SSM angle, θ, at the image time, t, from θ_n, t_n, θ_{n+1}, and t_{n+1}.

   $\theta = \theta_n * (t_{n+1} - t)/(t_{n+1} - t_n) + \theta_{n+1} * (t - t_n)/ (t_{n+1} - t_n)$

3. Compute the SSM reflection matrix.

$$\mathbf{M}_{SSM}(\theta) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta\cos\theta & \cos^2\theta & \sin\theta \\ \sin^2\theta & -\sin\theta\cos\theta & \cos\theta \end{bmatrix}$$

4. Apply the SSM reflection matrix to the LOS.
   [los$_{TIRS}$] = M$_{SSM}$(θ) [los$_{SSM}$]

a.3) Find Attitude

This function finds the precise roll, pitch and yaw at the specified time. This routine uses the "corrected" version of the attitude data stored in the TIRS LOS model. This attitude data sequence includes the effects of ground control point precision correction (if any).

Find the current time relative to attitude data start time stored in the LOS model.

dtime = time + image epoch time – attitude epoch time

Note:

time = nominal time of input sample relative to the start of the image epoch time = image start time from LOS model, only need seconds of day field since all epochs are adjusted to the same day.
attitude epoch time = attitude data start time from LOS model, only need seconds of day field since all epochs are adjusted to the same day.

Find index into attitude data (stored in model) corresponding to dtime:

$$\text{index} = \text{floor}\left(\frac{\text{dtime}}{\text{attitude sampling rate}}\right)$$

where:
attitude sampling rate = sample period from LOS model

This attitude index determination could also be implemented as a search through the attitude data time stamps, which are stored in the LOS model. The selected index would be the index of the last time that does not exceed dtime.

Attitude is found by linearly interpolating between the attitude values located at index and index+1 using the corrected attitude sequence from the LOS model:

$$w = \frac{\text{fmod}\left(\text{dtime, attitude sampling rate}\right)}{\text{attitude sampling rate}}$$

$$\text{roll} = \text{model roll}_{index} + \left(\text{model roll}_{index+1} - \text{model roll}_{index}\right) * w$$

$$\text{pitch} = \text{model pitch}_{index} + \left(\text{model pitch}_{index+1} - \text{model pitch}_{index}\right) * w$$

$$\text{yaw} = \text{model yaw}_{index} + \left(\text{model yaw}_{index+1} - \text{model yaw}_{index}\right) * w$$

a).3.i. Find Jitter Find the high frequency roll, pitch and yaw corrections at the specified input image line/sample coordinate. This routine uses the jitter table stored in the TIRS LOS model. This table is time aligned with the TIRS image line sampling times, so the jitter table look-up proceeds directly from the input line/sample coordinates:

Find the current detector number from the input sample location:

detector = round(sample)

Verify that the detector is in the valid range for this band (return error if not).

Look up the number of L0R fill pixels for this detector (from the fill table).

Calculate the jitter table index:

Index = round(line) – l0r_fill_pixels

Verify that jitter table index is within the valid range for the table (return zeros if not).

Extract the roll-pitch-yaw jitter values for the current index from the jitter table and return these values.

Note that the jitter values are a direct look-up without interpolation. This does not compromise accuracy because this function is only used for cases of EXACT detector projection (e.g., the TIRS data simulator) for which the input line/sample coordinates are integers. The jitter values extracted by Find Jitter are added to the low frequency roll-pitch-yaw values interpolated by Find Attitude, by the calling procedure, Project TIRS LOS, when the EXACT option is in force.

a.4) Move Satellite Sub-Algorithm

This function computes the satellite position and velocity at a delta time from the ephemeris reference time using Lagrange interpolation. This is a utility sub-algorithm that accesses the "corrected" version of the model ephemeris data to provide the TIRS position and velocity at any specified time. Since the model ephemeris arrays are inputs to this sub-algorithm it will work with either the ECI or ECEF ephemeris data.

Calculate time of current line/sample relative to start time of ephemeris start time.

reference time = time + image epoch time – ephemeris epoch time

where:

time = nominal time of input sample relative to the start of the imagery

image epoch time = image start time from LOS model, only need seconds of day since all epochs are on same day.

ephemeris epoch time = ephemeris start time from LOS model, only need seconds of day since all epochs are on same day.

Find index into ephemeris data stored in model.

$$\text{index} = \text{floor}\left(\frac{\text{reference time}}{\text{ephemeris time steps}} - \frac{\text{number of Lagrange points}}{2} \right)$$

where:

ephemeris time steps = time between ephemeris samples
number of Lagrange points = number of points to use in Lagrange interpolation

Use Lagrange interpolation to calculate satellite position and velocity in ECEF (or ECI, depending on which sequence is provided) coordinates at time of current line/sample.

X = Lagrange(model satellite ECEF/ECI x[index])
Y = Lagrange(model satellite ECEF/ECI y[index])
Z = Lagrange(model satellite ECEF/ECI z[index])
XV = Lagrange(model satellite ECEF/ECI vx[index])
YV = Lagrange(model satellite ECEF/ECI vy[index])
ZV = Lagrange(model satellite ECEF/ECI vz[index])

where:

X = satellite x coordinate
Y = satellite y coordinate
Z = satellite z coordinate
XV = satellite x velocity
YV = satellite y velocity
ZV = satellite z velocity

a.5) Convert Sensor LOS to Geocentric

This function finds the line of sight vector from the spacecraft to a point on the ground by transforming the line of sight vector in sensor coordinates to perturbed spacecraft coordinates.

Use the TIRS alignment matrix in the TIRS LOS model to convert the TIRS LOS vector from sensor to body. Then apply roll, pitch, and yaw to the LOS to convert body to orbital. Finally, use the ephemeris to construct the orbital to ECEF rotation matrix and use it to transform LOS to ECEF.

First, using the 3x3 ACS to instrument alignment transformation matrix stored in the TIRS LOS model, calculate the instrument to ACS transformation matrix.

$$[\text{Instrument to ACS}] = [\text{ACS to Instrument}]^{-1}$$

Transform LOS from Instrument to ACS/body coordinates.

$$[\text{navigation los}] = [\text{Instrument to ACS}]\ [\text{los}]$$

Calculate attitude perturbation matrix using interpolated attitude values. Note that these values include the effects of precision LOS correction (if any) as these will be built into the "corrected" attitude stream in the LOS model. Also note that for Earth-view acquisitions the roll-pitch-yaw values will be with respect to the orbital coordinate system but for celestial acquisitions they will be with respect to ECI.

Calculate perturbation matrix, [perturbation], due to roll, pitch, and yaw:

[attitude perturbation] = [Y_{yaw}] [P_{pitch}] [R_{roll}] =

$$\begin{bmatrix} \cos(p)\cos(y) & \sin(r)\sin(p)\cos(y)+\cos(r)\sin(y) & \sin(r)\sin(y)-\cos(r)\sin(p)\cos(y) \\ -\cos(p)\sin(y) & \cos(r)\cos(y)-\sin(r)\sin(p)\sin(y) & \cos(r)\sin(p)\sin(y)+\sin(r)\cos(y) \\ \sin(p) & -\sin(r)\cos(p) & \cos(r)\cos(p) \end{bmatrix}$$

Calculate new LOS in orbital coordinates (Earth-view) or ECI (celestial) due to attitude perturbation:

[perturbation los] = [perturbation] [navigation los]

For Earth-view acquisitions, calculate the transformation from Orbital Coordinates to ECEF. The position and velocity vectors used in calculating the transformation are those calculated above. These vectors are in ECEF allowing the LOS to be transformed from the instrument coordinate system to the ECEF coordinate system.

Transform perturbed LOS from Orbital to ECEF.

[ECEF los] = [ORB2ECEF] [perturbation los]

For celestial acquisitions, the ECI los ([perturbation los]) is returned.

a.6) Find Target Position

This function finds the position where the line of sight vector intersects the Earth's surface.

Intersect the LOS in ECEF with the Earth model calculating the target ECEF vector. The ECEF vector is then used to compute the geodetic latitude and the longitude of the intersection point (see Figure 4-67).

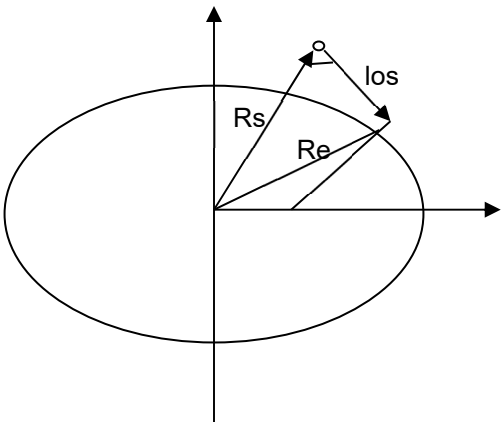

Figure 4-67. Intersecting LOS with Earth Model

Where:

Rs = satellite position vector
Re = geocentric Earth vector
los = line-of-sight vector

Intersect LOS with ellipsoid

1. Rescale vectors with ellipsoid parameters.

$$Rs' = \left[\frac{Rs_x}{a} \quad \frac{Rs_y}{a} \quad \frac{Rs_z}{b}\right]$$

$$Re' = \left[\frac{Re_x}{a} \quad \frac{Re_y}{a} \quad \frac{Re_z}{b}\right]$$

$$los' = \left[\frac{los_x}{a} \quad \frac{los_y}{a} \quad \frac{los_z}{b}\right]$$

where:

a = semi-major axis of Earth ellipsoid
b = semi-minor axis of Earth ellipsoid
Rs' = rescaled satellite position vector
Re' = rescaled geocentric Earth vector
los' = rescaled LOS vector

2. From the Law of Cosines

$$|Re'|^2 = |d * los'|^2 + |Rs'|^2 - 2|d * los'||Rs'|\cos(\delta)$$

where:

d = los' vector length
δ = angle between Rs' and los'

$$\cos(\delta) = \frac{los' \bullet Rs'}{|los'||Rs'|}$$

By definition | Re' | = 1

Rearranging the equation determined from the Law of Cosines in terms of the constant d.

$$d^2|los'|^2 + 2d(los' \bullet Rs') + |Rs'|^2 - 1 = 0$$

Solving for d using the quadratic equation.

$$d = \frac{-(los' \bullet Rs') - \sqrt{(los' \bullet Rs')^2 - |los'|^2(|Rs'|^2 - 1)}}{|Rs'|^2}$$

3. Compute new target vector.

$$Re' = Rs' + d * los'$$

4. Rescale target vector.

$$Re = \begin{bmatrix} a * Re'_x & a * Re'_y & b * Re'_z \end{bmatrix}$$

5. Compute Geodetic coordinates (see Geocentric to Geodetic below).

(Re'$_x$, Re'$_y$, Re'$_z$) => (ϕ_0, λ_0, h_0)

If target height (H), or elevation corresponding to current z plane, is not zero:

Initialize:
Target vector: rt=Re
Target height: h_0=0

Iterate until $\Delta h = (h_i - H)$ is less than TOL

a) Calculate delta height.

$\Delta h = h_i - H$

b) Compute length of LOS.

$$d = \sqrt{(rtx - rsx)^2 + (rty - rsy)^2 + (rtz - rsz)^2}$$

where:

d = length of LOS vector
rt = target vector
rs = spacecraft position vector

c) Compute LOS /height sensitivity.

$$q = n \bullet los$$

Where **n** is a vector normal to the ellipsoid surface.

$$n = \left[\cos(\varphi_i)\cos(\lambda_i) \quad \cos(\varphi_i)\sin(\lambda_i) \quad \sin(\lambda_i)\right]^T$$

and:
q = LOS height sensitivity coefficient
los = LOS unit vector
ϕ_i = current estimate of ground point latitude
λ_i = current estimate of ground point longitude

d) Adjust LOS.

$$d = d + q * \Delta h$$

e) Re-compute target vector.

$$rt = rs + d * los$$

f) Calculate new geodetic coordinates and corresponding height above ellipsoid.

$$(rtx, rty, rtz) \Rightarrow (\phi_{i+1}, \lambda_{i+1}, h_{i+1})$$

Calculate the geodetic latitude and longitude from the final ECEF vector.

a.7) Geocentric to Geodetic

The relationship between ECEF and geodetic coordinates can be expressed simply in its direct form:

$e^2 = 1 - b^2 / a^2$
$N = a / (1 - e^2 \sin^2(\varphi))^{1/2}$
$X = (N + h) \cos(\varphi) \cos(\lambda)$
$Y = (N + h) \cos(\varphi) \sin(\lambda)$
$Z = (N (1-e^2) + h) \sin(\varphi)$

where:

X, Y, Z	-	ECEF coordinates
φ, λ, h	-	Geodetic coordinates (lat □, long □, height h)

N	-	Ellipsoid radius of curvature in the prime vertical
e^2	-	Ellipsoid eccentricity squared
a, b	-	Ellipsoid semi-major and semi-minor axes

The closed-form solution for the general inverse problem (which is the problem here) involves the solution of a quadratic equation and is not typically used in practice. Instead, an iterative solution is used for latitude and height for points that do not lie on the ellipsoid surface, i.e., for h ≠ 0.

To convert ECEF Cartesian coordinates to spherical coordinates:

Define:

$$\text{radius} = \sqrt{X^2 + Y^2 + Z^2}$$

$$\varphi' = \sin^{-1}\left(\frac{Z}{\text{radius}}\right)$$

$$\lambda = \tan^{-1}\left(\frac{Y}{X}\right)$$

Initialize:

$$\theta = \varphi'$$

$$h_0 = 0$$

Iterate until abs(h_i-h_{i+1}) < TOL

$$\text{re} = \frac{a * \sqrt{1-e}}{\sqrt{1-e * \cos^2(\theta)}}$$

$$\varphi = \tan^{-1}\left(\frac{\tan(\theta)}{1-e}\right)$$

$$\Delta\varphi = \varphi - \theta$$

$$\text{rs} = \text{radius}^2 - \text{re}^2 * \sin^2(\Delta\varphi)$$

$$\text{h}_{i+1} = \sqrt{\text{rs}} - \text{re} * \cos(\Delta\varphi)$$

$$\theta = \varphi' - \sin^{-1}\left(\frac{\text{h}_{i+1}}{\text{radius} * \sin(\Delta\varphi)}\right)$$

Projection Transformation

This function converts coordinates from one map projection to another. The transformation from geodetic coordinates to the output map projection depends on the type of projection selected. The mathematics for the forward and inverse transformations for the UTM, Lambert Conformal Conic, Transverse Mercator, Oblique Mercator, Polyconic, and the Space Oblique Mercator (SOM) map projections are handled by USGS's General Cartographic Transformation Package (GCTP), as noted below.

Projection Errors
This function reports projection transformation package errors. The function receives a GCTP error code and prints the correct error message.

General Cartographic Transformation Package (gctp)
Map projections are handled by USGS's General Cartographic Transformation Package (GCTP).

Grid Structure Summary
Table 4-37 and Table 4-38 below show the detailed contents of the TIRS LOS grid structure.

Geometric Grid Structure Contents
Satellite Number (8)
WRS Path
WRS Row (may be fractional)
Acquisition Type (Earth, Lunar, Stellar)
Scene Framing Information:
Frame Type: PROJBOX, MINBOX, MAXBOX, PATH, PATH_MINBOX, PATH_MAXBOX
Projection Units (text): METERS, RADIANS, ARCSECONDS
Projection Code: GCTP integer code for UTM, SOM, etc...
Datum: WGS84
Spheroid: GCTP integer code = 12 (WGS84/GRS80)
UTM Zone: UTM zone number (or 0 if not UTM)
Map Projection Parameters: 15-element double array containing parameters
Corners: 4 by 2 array of projection coordinates for UL, LL, UR, and LR corners
Path-oriented Framing Information:
Center Point: latitude and longitude of WRS scene center
Projection Center: Map x/y of WRS scene center
Rotation Angle: Rotation (from true north) of the path frame (degrees)
Orientation Angle: Rotation (from grid north) of the path frame (degrees)
Active Image Areas: latitude and longitude (in degrees) of the four corners of the active image area (excluding leading and trailing SCA imagery) for each band
Grid Structure Information:
Number of SCAs
Number of Bands
Band List: array of band IDs included in grid
Array of band grid structures, one for each SCA in each band (see Table 4-38)

Table 4-37. TIRS LOS Grid Structure Contents

Grid Structure Contents for Each SCA in Each Band
Band number
Grid cell size: number of image lines and samples in each grid cell
Grid cell scale: 1/lines per cell and 1/samples per cell
Pixel size: in projection units (usually meters)
Number of lines in output image
Number of samples in output image

Grid Structure Contents for Each SCA in Each Band
Number of lines in grid (NL)
Number of samples in grid (NS)
Number of z-planes (NZ)
Index of zero-elevation z-plane
Z-plane spacing: elevation increment between z-planes
1D array of input line numbers corresponding to each grid row
1D array of input sample numbers corresponding to each grid column
3D array of output lines for each grid point (row-major order) (NS*NL*NZ)
3D array of output samples for each grid point (row-major order) (NS*NL*NZ)
Array of line c_0, c_1 even/odd offset coefficients (row-major order) (2*NS*NL)
Array of sample d_0, d_1 even/odd offset coefficients (row-major order) (2*NS*NL)
3D array of forward mapping (ils2ols) coefficient sets (NS*NL*NZ)
3D array of inverse mapping (ols2ils) coefficient sets (NS*NL*NZ)
3D array of line jitter sensitivity coefficient vectors (3*NS*NL*NZ)
3D array of sample jitter sensitivity coefficient vectors (3*NS*NL*NZ)
Degree of rough polynomial
Array of rough line polynomial coefficients ((degree+1)^2 * NZ values)
Array of rough sample polynomial coefficients ((degree+1)^2 * NZ values)

Table 4-38. Per Band LOS Grid Structure Contents

LOS Projection Grid Size

To fully capture the potential variability of the 50 Hz attitude data would require a grid spacing of 1.4 TIRS lines. This is impractical. The TIRS error budgets assumed that attitude variations at frequencies up to 10 Hz would be corrected in the LOS model. Such variations can be captured by sampling at 20 Hz or higher. This corresponds to a grid spacing of 3.5 lines, which is still not terribly practical. A nominal grid spacing of 5 lines was initially adopted for TIRS. Grid size is not the concern it is for OLI data as the TIRS images are substantially smaller (only 2 bands, 3 SCAs and ~2071 lines) but the practicalities of working with such a dense grid (e.g., on the grid cell search logic) make it desirable to implement per-line high-frequency correction logic as this permits the use of a sparser grid. The inclusion of a high frequency jitter table in the TIRS model and jitter sensitivity coefficients in the grid structure allow the grid to be less dense in the time (line) dimension. The baseline assumption is that attitude frequencies above 1 Hz will be relegated to the jitter table allowing the TIRS grid density to be reduced to 10 lines thus saving grid space even with the addition of the new jitter sensitivity fields.

4.3.2.7 Notes

Some additional background assumptions and notes include the following:

1. The NOVAS planetary ephemeris file provides the lunar ephemerides used to define the reference output space for lunar image processing. This file is in the original JPL format and is provided to the NOVAS routines as an input.
2. The number of elevation planes in the grid is computed from the elevation range provided by the DEM and the maximum elevation plane spacing stored in the CPF.

3. The default grid density is hard coded (through #define statements) but these values are overwritten by values read from the CPF.
4. The "thresholds and limits" parameters, stored either in system tables or the database for L7 and ALI, have been moved to the CPF. This makes date specific changes, e.g., due to a change in the nominal orbit during early- or late-mission operations, easier to manage.
5. The problem of multiple terrain intersections needs to be addressed for off-nadir images, though probably not for purposes of grid generation, since it requires analysis of the full resolution DEM. This problem is being handled for OLI images by a new processing step, which creates an output space mask of pixels that are obscured by terrain. This is not as much of a concern for the (lower resolution) TIRS images and is ignored. Refer to the Terrain Occlusion Algorithm for details.
6. A number of data elements that are shown as coming from the TIRS LOS Model (e.g., TIRS to ACS reference alignment matrix/quaternion, spacecraft CM to TIRS offset in ACS reference frame, focal plane model parameters) are also contained in and were loaded into the model from the CPF. This is a departure from the ALIAS prototype, which accesses fields of this type from the CPF wherever needed rather than merging them into the TIRS LOS model.

4.3.3 TIRS Resampling Algorithm

4.3.3.1 Background/Introduction

Since the TIRS uses a pushbroom sensor architecture that is very similar to the OLI, we can adopt a common approach to image resampling. Despite the common approach, the TIRS resampling algorithm was documented in a separate algorithm description, reflecting the substantially different OLI and TIRS algorithm development schedules. The ultimate operational implementations of the OLI and TIRS resampling algorithms were combined into a single application.

The resampling algorithm is used to take a L1R image in original sensor geometry, which has unevenly spaced pixels with respect to the surface of the object imaged, and creates a reprojected image where all image pixels are located within an evenly spaced set of grid points, or output space, with respect the object imaged. This mapping is subject to the errors associated with the interpolation method used to determine the digital numbers associated with the output image.

The TIRS geometric resampling grid and geometric model are used to calculate the mappings between the input and output space. The TIRS geometric model contains the individual detector offsets from a nominal location (i.e., departures from the Legendre polynomial line-of-sight model, offsets due to bad detector replacement) while the geometric resampling grid contains all other mapping variables. The resampling grid provides a mapping from a 2D input space to a 3D output space and vice versa. The output space corresponds to x/y/z projection locations while the input space corresponds to line/sample locations within the L1R. The z component in output space is elevation. If elevation is not to be accounted for during processing an elevation of zero is used for mapping output pixels to input pixels.

Due to what can be rather large sample-to-sample offsets within a L1R image, the cubic convolution interpolation option works in a two-step process. A hybrid set of pixels in the sample direction are created using cubic convolution resampling in the line direction. This creates a set of unevenly spaced pixels in the sample direction. The Akima A interpolation method is then used to determine the final digital number for the output image by resampling the hybrid pixels in the sample direction. The nearest neighbor resampling option simply determines the closest input pixel for corresponding output pixel.

The TIRS resampling algorithm is derived from the corresponding OLI algorithm. The sensor architecture between the instruments is similar enough that a majority of the OLI algorithm can be reused. Though the TIRS focal plane geometry is somewhat simpler than the OLI due to the lack of detector stagger, it can be represented as a special case of the same model through appropriate selection of calibration parameters. The baseline geometric modeling approach will use the same 3D gridding approach for OLI and TIRS.

The USGS defines products L1G, radiometrically calibrated with only systematic geometric corrections, L1GT, radiometrically calibrated systematic corrections and the use of a Digital Elevation Model (DEM) to correct for relief displacement, and L1TP, radiometrically calibrated and orthorectified using ground control points and a DEM to correct for relief displacement. Where appropriate this algorithm will specifically designate the type of product to which a section of text applies. If the reference applies to all three product types then the designation L1G will be used.

4.3.3.2 Dependencies

The OLI/TIRS resampling algorithm assumes that the Ancillary Data Preprocessing, OLI and TIRS LOS Model Creation, and OLI and TIRS Line-of-Sight Projection to Ellipsoid and Terrain algorithms have been executed, and a L1R has been generated. If a digital elevation model (DEM) is given as input to account for relief, or terrain, displacement the grid must have an adequate number and range (elevation bounds) of z-planes to cover the entire elevation range within the L1R. A geometric model and grid must be available for the L1R. More information about the data structure and contents of the Geometric Model and Resampling Grid can be found in the Ancillary Data Preprocessing, OLI LOS Model Creation, OLI LOS Projection, TIRS LOS Model Creation, and TIRS LOS Projection Algorithms (Sections 4.1.4, 4.2.1, 4.2.2, 4.3.1, and 4.3.2).

4.3.3.3 Inputs

The resampling algorithm and its component sub-algorithms use the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs
L1R Image
TIRS Resampling Grid (see the TIRS Line of Sight Projection Algorithm for contents)

Algorithm Inputs
Bands to process
Terrain correction Flag (yes/no)
DEM (if terrain flag set to yes)
SCA combine flag (yes/no)
TIRS Geometric Model (see TIRS Line of Sight Model Creation Algorithm for contents)
Resampling type (CC,NN)
Minimum and maximum DN of output
Output data type
α (if resampling type is CC) (defaults to -0.5)
Fill pixel value

4.3.3.4 Outputs

Resampled output image (L1G, L1GT or L1TP)
Image data descriptor record (DDR) (see Table 4-39)

4.3.3.5 Options

Cubic convolution or nearest neighbor resampling
Creating an output image with Sensor Chip Assemblies (SCAs) combined or separated
Applying terrain correction, yes or no

4.3.3.6 Procedure

TIRS resampling interpolates radiometrically corrected but geometrically raw image data to a map projected output space. The resampling process uses information stored in the TIRS resampling grid along with focal plane calibration data stored in the TIRS geometric model to map output projection locations to an input location. Since an input location for an output pixel typically lies at a non-integer location interpolation is used to find the pixel values associated with this non-integer location. TIRS resampling is performed on the geometrically raw L1R data using one of two methods; cubic convolution combined with the Akima A method, or nearest neighbor. Note that modulation transfer function compensation (MTFC) and bilinear resampling are not supported in the baseline algorithm, although MTFC has been implemented as a separate pre-process (see Section 4.3.7). Due to the lack of inherent band registration and the need to perform subpixel registration to achieve TIRS band alignment, cubic convolution combined with the Akima A interpolation method will be used to generate the standard L8/9 products. It is also important to have the best subpixel accuracy in the output image during geometric characterization and calibration, so cubic convolution is chosen for interpolation during the characterization and calibration of the TIRS instrument. The ALIAS-heritage nearest neighbor interpolation capability is also provided as an option for special-purpose science products and testing purposes. Since both standard product generation and geometric characterization and calibration are the focus of this document, the only interpolation method discussed in detail here is the cubic convolution combined with the Akima A method.

During resampling, there is a need to know what input pixel goes with a given output pixel. The geometric processing system does not have a “true” inverse model to

perform this calculation. Instead, for a given output pixel, the corresponding input pixel is found from the forward and inverse mapping coefficients stored in the resampling grid. There are two scenarios when performing this calculation. The first involves performing resampling for a systematic image in which case the dimension for z, or elevation, is zero. This involves only a two dimensional operation in line and sample. The second involves performing resampling for a terrain corrected image. A terrain corrected image has the effects of relief removed from the output imagery. When working with a terrain-corrected image, a 3-dimensional operation is performed during the inverse mapping with the dimensions being input (L1R) line, input sample, and elevation (see Figure 4-68). Both procedures of mapping output pixel locations to input pixel locations are discussed below.

Due to the layout of the TIRS focal plane, there are along-track offsets between spectral bands within each SCA, along-track offsets between even and odd SCAs, and a reversal of the band ordering in adjacent SCAs. This leads to an along-track offset in the imagery coverage area for a given band between odd and even SCAs as well as an offset between bands within each SCA. To create more uniform image coverage within a geometrically corrected output product, the leading and trailing imagery associated with these offsets is trimmed. This trimming is controlled by a set of latitude/longitude bounds for the active image area for each band, contained in the input resampling grid. Trimming is implemented by converting these bounds to a look up table that lists the starting and ending sample location of active (non-fill) data for each line of the output image.

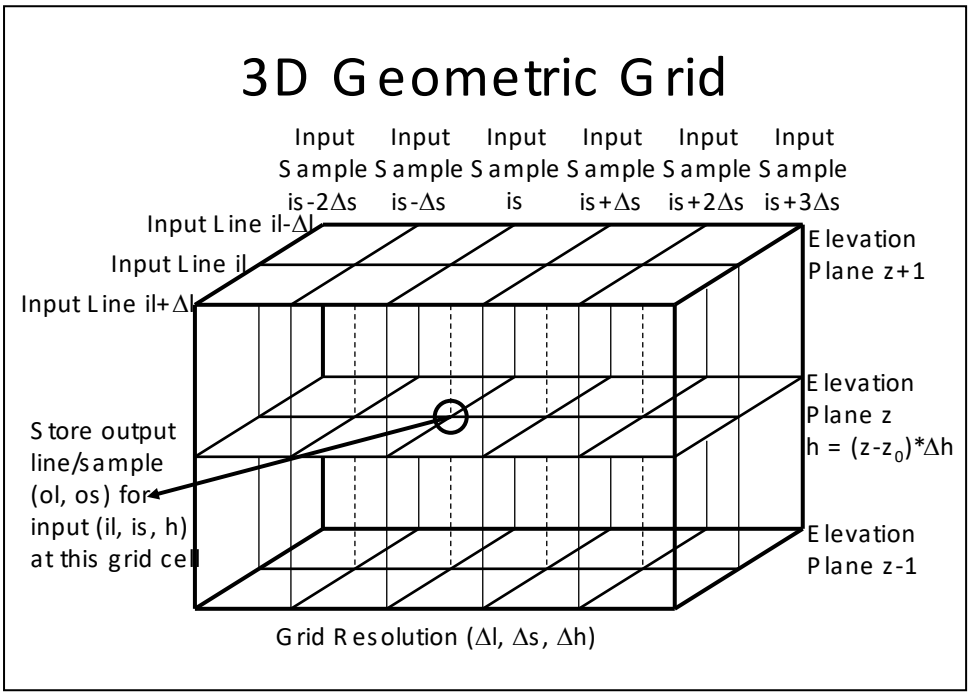

Figure 4-68. 3D Grid Representation

Using the geometric grid to map an output pixel location to an input pixel location
To find an input line/sample location for an output line/sample location given that the elevation is zero:

1) Calculate an input line and sample location using the rough polynomial stored in the resampling grid and the current output line and sample location.

$$\text{approximate input line} = \sum_{n=0}^{N}\left[\sum_{m=0}^{M}\left(ra_{n,m} * (\text{output sample})^{m}\right) * (\text{output line})^{n}\right]$$

$$\text{approximate input sample} = \sum_{n=0}^{N}\left[\sum_{m=0}^{M}\left(rb_{n,m} * (\text{output sample})^{m}\right) * (\text{output line})^{n}\right]$$

Where:

ra = rough polynomial mapping coefficients for line mapping
rb = rough polynomial mapping coefficients for sample mapping
M = Number of sample coefficients in polynomial
N = Number of line coefficients in polynomial

Previous experience when working with the ALI instrument has demonstrated a 1st order polynomial in both the line and sample direction will suffice for the rough polynomial, thus M = N = 1.

$$\text{approximate input line} = a_0 + a_1 * \text{output sample} + a_2 * \text{output line} + a_3 * \text{output sample} * \text{output line}$$

$$\text{approximate input sample} = b_0 + b_1 * \text{output sample} + a_2 * \text{output line} + a_3 * \text{output sample} * \text{output line}$$

There is no evidence to believe that this will not also be the case when working with the TIRS instrument.

2) Calculate the grid cell location for the approximate input line and sample location.

$$\text{row} = \frac{\text{approximate input line}}{\text{number of lines per cell}}$$

$$\text{column} = \frac{\text{approximate input sample}}{\text{number of samples per cell}}$$

Where:
number of lines per cell = size of grid cell in lines
number of samples per cell = size of grid cell in samples

Set this grid cell column and row location as the current grid cell column and row location.

3) Using the current grid cell location check if the correct grid cell has been found.

Use input (current) mapping grid cell coefficients (a_i and b_i) to map output line and sample to input:

input line = $b_0 + b_1$ * output sample + b_2 * output line + b_3 * output line * output sample
input sample = $a_0 + a_1$ * output sample + a_2 * output line + a_3 * output line * output sample

Calculate the grid cell location for this input line and sample location:

$$\text{new row} = \frac{\text{input line}}{\text{number of lines per cell}}$$

$$\text{new column} = \frac{\text{input sample}}{\text{number of samples per cell}}$$

If the new grid cell (new row and new column) is the same as the current grid cell (current row and current column):

The correct grid cell has been found, inverse grid mapping coefficients for this grid cell are used to calculate the input line/sample for the current output line/sample.

If the new grid cell (new row and new column) is not the same the current grid cell (current row and current column):

The new grid cell is chosen as current grid cell and the 3rd step is repeated until the correct grid cell is found.

This routine or function listed above, of mapping output pixel locations to input pixel locations without taking into account elevation, will be referred to as ols2ils (output space line-sample to input space line-sample mapping). The ols2ils sub-algorithm takes a given output line and sample location and calculates the grid cell column and row location along with the corresponding input line and sample location for that output location.

To find an input line/sample location for an output line/sample location given that the elevation is not zero:

1. Find the z planes that the elevation associated with the output pixel falls between.

$$z\ \text{plane} = (\text{int})\text{floor}\left(\frac{\text{elevation}}{\text{elevation increment}}\right) + z_{elev=0}$$

Where:
elevation = elevation associated with current output location (from DEM)
elevation increment = elevation increment between z planes stored in grid
$z_{elev=0}$ = zero z plane, the index of the zero elevation z-plane

The output line/sample falls between z plane and z plane+1.

2. Call ols2ils for z plane and z plane+1. This yields (input $sample_0$, input $line_0$), and (input $sample_1$, input $line_1$).
3. Interpolate between z plane and z plane + 1 to find input line and sample location for elevation.

Calculate elevations for z plane and z plane + 1:
$elev_0$ = elevation increment * (z plane - zero z plane)
$elev_1$ = $elev_0$ + elevation increment

Calculate weights for ols2ils results:

$$w_0 = \frac{elev_1 - elevation}{elev_1 - elev_0}$$

$$w_1 = \frac{elevation - elev_0}{elev_1 - elev_0}$$

input sample = input $sample_0$ * w_0 + input $sample_1$ * w_1
input line = input $line_0$ * w_0 + input $line_1$ * w_1
Where:
input $sample_0$ = input sample for z plane
input $sample_1$ = input sample for z plane + 1
input $line_0$ = input line for z plane
input $line_1$ = input line for z plane + 1

This routine or function listed above, which performs the three-dimensional output space line-sample to input space line-sample mapping, is referred to as 3d_ols2ils.

Resampling Methodology
The along and cross track detector offsets are applied during resampling. These include both the dynamic primary and redundant detector terrain-dependent relief and parallax effects that were calculated during the resampling grid generation, and the individual detector shifts that are stored in the TIRS geometric model. The nature of these geometric effects due to the individual detector characteristics is such that, in input space, they are evenly spaced in the line direction but unevenly spaced in the sample direction. This is because as you move along raw imagery in the line direction, the detector number does not change. Since the detector number does not change along the line direction in raw input space, the along-track detector offset, stored within the geometric model, does not change. These geometric effects, due to these detector offsets, are slowly varying in time staying essentially constant within the area that resampling is performed. Therefore the along-track geometric effect, and essentially spacing in the line direction, can be treated as a constant over this area. The same logic helps explain why the across-track detector offset is not constant in the sample direction, since each sample comes from a different detector. This creates unevenly spaced samples in raw input space. An example of a detector layout and its associated offset can be seen in Figure 4-69. The squares in Figure 4-69 represent a location of an input pixel, taking into account the detector offsets. The circle with the cross-hairs in

Figure 4-69 represents the true input location for the current output pixel. It is at this point that an interpolated value is needed to represent the current output pixel.

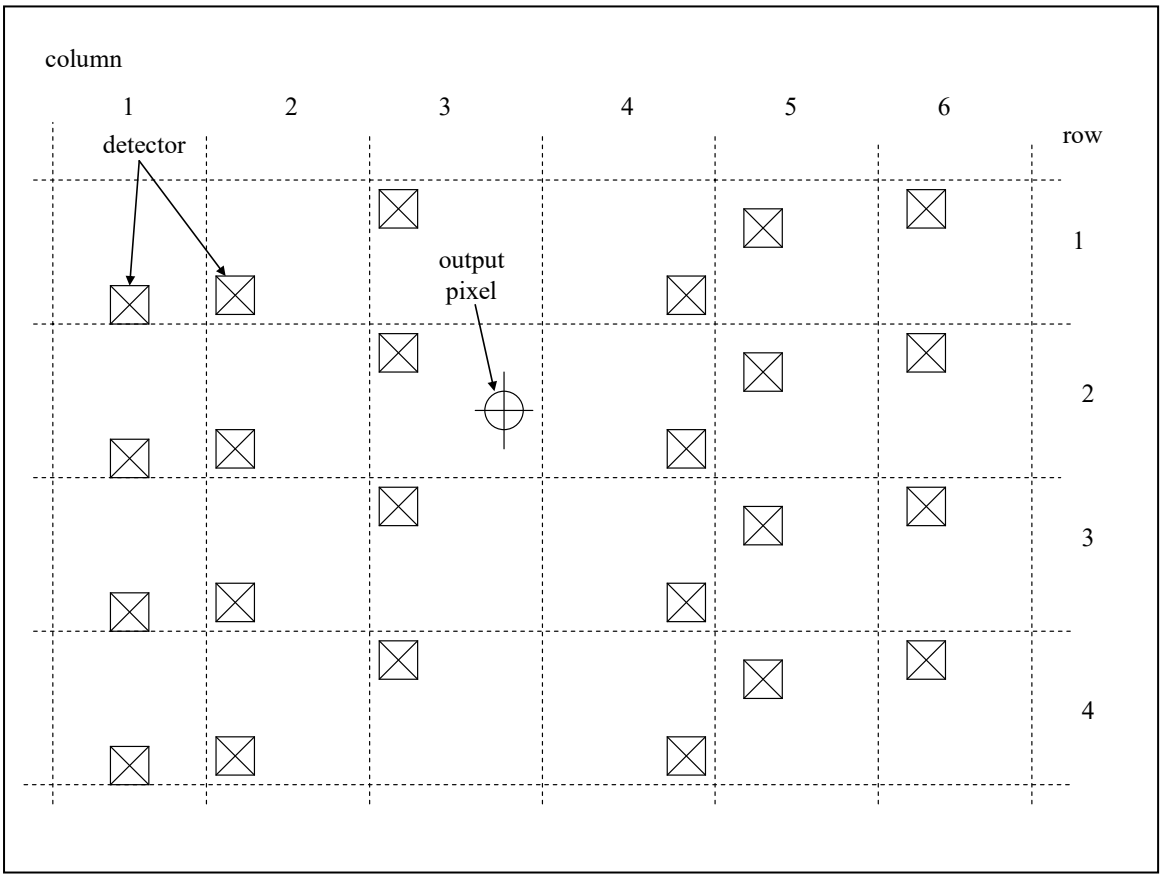

Figure 4-69. Example Detector Layout

Detector offsets are handled in the resampler by first applying a resampling kernel in the line direction that assumes evenly spaced detectors. Cubic convolution interpolation is used in the line direction; this will align a set of pixels in the sample direction. Once the pixels are aligned in the sample direction, at uneven spacing, the Akima A interpolation is used to find the final output pixel value. The linear arrangement of the TIRS detectors, not accounting for bad detector replacement, may have made it possible to avoid this complication, but treating the TIRS detectors as a special case with zero even/odd detector stagger, allows the use of a common resampling approach for OLI and TIRS.

Cubic convolution interpolation uses a set of piecewise cubic spline interpolating polynomials. The polynomials have this form:

$$f(x)=\begin{cases}(\alpha+2)|x|^3-(\alpha+3)|x|^2+1 & 0\le|x|<1\\ \alpha|x|^3-5\alpha|x|^2+8\alpha|x|-4\alpha & 1\le|x|<2\\ 0 & |x|>2\end{cases}$$

Four points, centered on the point to be interpolated, are used in interpolation. The weights for each point are generated from *f(x)*. The α in the cubic convolution function is a variable parameter that effects the edge slope of the function. For standard processing, a value of -0.5 is used. An example of what the cubic convolution function looks like, and the corresponding weights for a phase shift of zero (marked as x's), is shown in Figure 4-70.

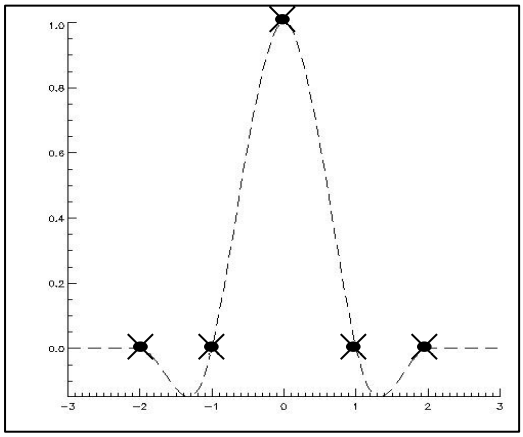

Figure 4-70. Cubic Convolution Function

As stated previously; for the TIRS resampler the cubic convolution resampling process produces a set of hybrid points that are aligned in the line direction. This is done by resampling several sets of L1R pixels in the line direction using the cubic convolution kernel; each time cubic convolution is performed one hybrid pixel is produced. The set of hybrid points produced from the cubic convolution process are not evenly spaced in the sample direction. Figure 4-71 illustrates a set of hybrid samples that have been aligned in the line direction using the cubic convolution process.

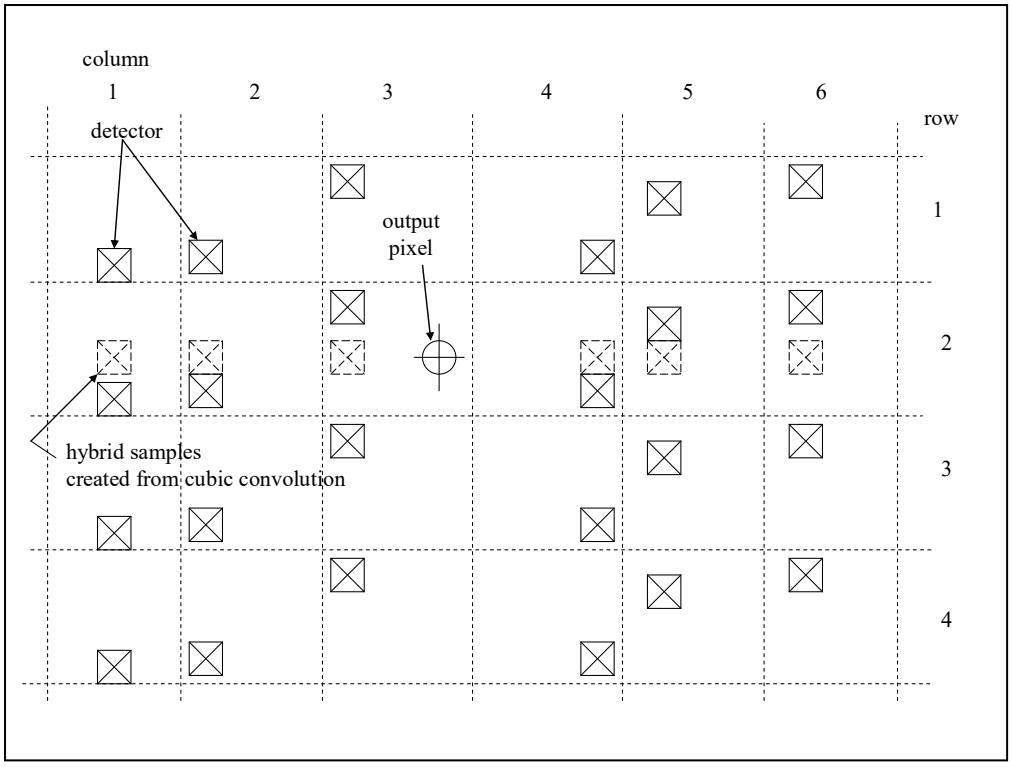

Figure 4-71. Hybrid Pixels for Detector Offsets

The Akima A method for interpolation is used for interpolating the hybrid pixels created from the cubic convolution process. This method of interpolation does not require the samples used to be evenly spaced. The Akima A method uses a third order polynomial for interpolation. The interpolating polynomial is defined by the coordinates and the slopes of the two points that are on either side of the point to be interpolated. The slopes of the adjacent points are determined as follows:

If five points are defined as 1, 2, 3, 4, and 5 then the slope at point 3, t, is defined as:

$$t = \frac{|m_4 - m_3|m_2 + |m_2 - m_1|m_3}{|m_4 - m_3| + |m_2 - m_1|}$$

Where:

m_1 = slope of line segment defined by points 1 and 2
m_2 = slope of line segment defined by points 2 and 3
m_3 = slope of line segment defined by points 3 and 4
m_4 = slope of line segment defined by points 4 and 5

The Akima A method of interpolation is based upon the values (y) and slopes (t) on either side of the point that is to be interpolated. The interpolating polynomial for a point x between x_i and x_{i+1} is then defined as:

$$y = y_i + t_i * (x - x_i) + \frac{3\frac{y_{i+1} - y_i}{x_{i+1} - x_i} - 2t_i - t_{i+1}}{x_{i+1} - x_i} * (x - x_i)^2 + \frac{t_i + t_{i+1} - 2\frac{y_{i+1} - y_i}{x_{i+1} - x_i}}{(x_{i+1} - x_i)^2} * (x - x_i)^3$$

Where:

x = sample location of point to be interpolated
x_i = location of point to the left of x
x_{i+1} = location of point to the right of x
y_i = DN value for the input point at x_i
y = interpolated DN value for an output line and sample location

This methodology must be adjusted somewhat to account for higher frequency image distortion effects than those that can be captured by the conventional resampling grid. To model such effects, the L8/9 attitude data stream is separated in to low-frequency and high-frequency segments with the low-frequency portion being used for the TIRS line-of-sight projection operations that build the resampling grid. The high-frequency data are interpolated to match the TIRS line sampling times and stored in the TIRS LOS model in a jitter table for application as an extra correction at image resampling time. The process of separating the attitude data stream by frequency is described in the TIRS Line-of-Sight Model Creation Algorithm (Section 4.3.1).

Sensitivity coefficients that relate these high-frequency roll-pitch-yaw jitter terms to the equivalent input image space line and sample offset effects are stored in the TIRS LOS grid. This makes it possible to look up the roll-pitch-yaw jitter for each image line being resampled, and convert the jitter values to compensating input line/sample corrections that are used to refine the image interpolation location coordinates. The generation of these sensitivity coefficients is described in the TIRS Line-of-Sight Projection/Grid Generation Algorithm (see Section 4.3.2). The process by which the jitter table from the TIRS model and jitter sensitivity coefficients from the TIRS grid are used during image resampling is shown schematically in Figure 4-72. The items in green in the figure are new structures, added to support jitter correction.

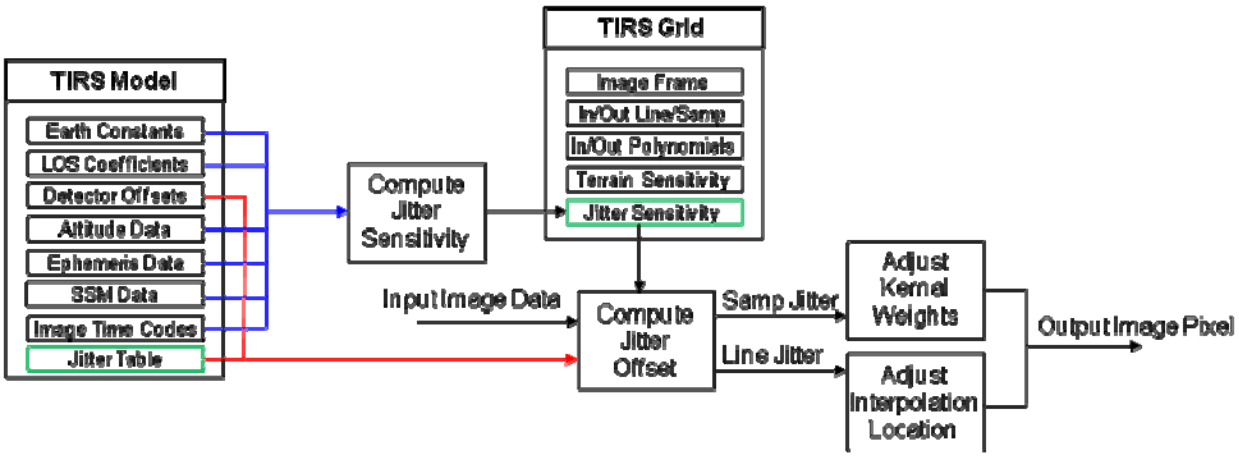

Figure 4-72. TIRS LOS Model and TIRS LOS Grid Jitter Correction Data Flow

Since the jitter effects vary by image line, the time delay between deselected detectors can lead to slightly different jitter effects in adjacent image samples. This is depicted in Figure 4-73. Six time samples (t0 through t5) for six adjacent detectors are shown in the figure. In this example, every other detector is assumed to be deselected and replaced from the redundant detector row. Note that the input line location returned by the grid is adjusted differently for the even and odd detectors due to their timing offset. Including the effects of detector deselect, the interpolated line location for the hybrid pixels could be different for each detector. The current approach does not account for sample-to-sample variations in jitter for each detector, applying the jitter correction only at the output location. This preserves the uniform along-track sampling assumption required to apply the cubic convolution kernel. Also note that while it is the interpolation location that is adjusted relative to the input pixel locations in the line direction, it is the detector sample locations that are adjusted relative to the interpolation location in the sample direction. The jitter-adjusted resampling procedure is explained in more detail below.

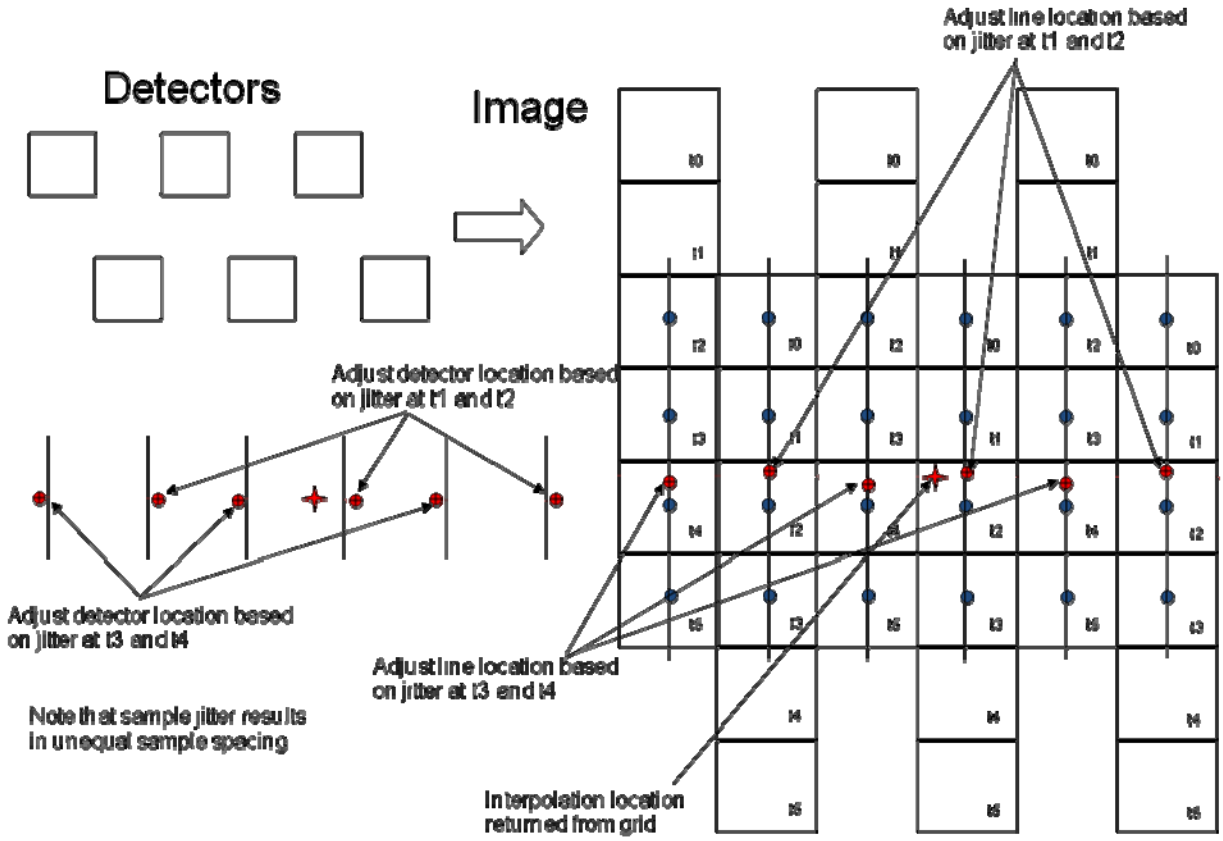

Figure 4-73. Jitter Effects in Image Resampling

4.3.3.6.1 Building The SCA-trimmed Look Up Table (LUT)

Allocate SCA-trim LUT. There is a starting and ending sample location of active or valid imagery stored for each line of output in the SCA-trimming look-up table.

LUT = malloc(2 * nl)

Where nl = number of lines in output imagery

Given the set of geographic corner coordinates, read from the input grid file, that represents valid imagery for a given band:

1. Map four corners to output projection coordinates.
2. Map four output projection coordinates to line and sample coordinates.
3. Set up polygon definition from four coordinates:
   <px0,py0> = <sample upper left, line upper left>
   <px1,py1> = <sample upper right, line upper right>
   <px2,py2> = <sample lower right, line lower right>
   <px3,py3> = <sample lower left, line lower left>
   <px4,py4> = <sample upper left, line upper left>
4. Set up sample locations for each line that is outside active imagery:
   osamp1 = -1.0
   osamp2 = output number of samples
   for nn = 0 to 3
   if px[nn] < osamp1 then osamp1 = px[nn] – 1.0
   if px[nn] > osamp2 then osamp2 = px[nn] + 1.0
5. Initialize LUT values to fill for all output lines:
   For nn = 0 to (2 * number of output lines)
   LUT[nn] = 0
6. For nn = 0 to number of output lines (nn and current line are synonymous).
   6.1. Define line by sample locations calculated from 4 and current line
   <x0,y0> = <osamp1, nn>
   <x1,y1> = <osamp2, nn>
   6.2. Determine intersection between sides of polygon defined in 3 and line defined in 6.1
   Initialize number of intersections for current line:
   intersections = 0
   For nn = 0 to 3
   (Simple line intersection routine)
   xlk = x0 – x1
   ylk = y0 – y1
   xnm = px[nn] – px[nn+1]
   ynm = py[nn] – py[nn+1]
   xmk = px[nn+1] – x1
   ymk = py[nn -1] – y1
   det = xnm * ylk – ynm * xlk
   if (| det | <= TOL) lines are parallel, no intersection found.
   s = (xnm * ymk - ynm * xmk) / det
   t = (xlk * ymk - ylk * xmk) / det
   if(s<0.0 || s>1.0 || t<0.0 || t>1.0)
   no intersection found
   else
   intersection found, calculate point:
   xp[intersections] = x1 + xlk * s
   yp[intersections] = y1 + ylk * s

intersections++

6.3. If number of intersections from 6.2. is two then the current line has valid active imagery and the look up table values are these intersections and represent the start and stop of valid imagery. Store values in SCA-trim lookup table.

```
if   xp[0] > xp[1]
        LUT[ 2 * nn ]      = xp[1]
        LUT[ 2 * nn + 1] = xp[0]
else
        LUT[ 2 * nn ]      = xp[0]
        LUT[ 2 * nn + 1] = xp[1]
```

(Note: If number of intersections is not two then current line has no valid active imagery and SCA-trim lookup table will contain points outside of imagery, fill will be used).

4.3.3.6.2 Load/Build Information

To resample a Level 1R dataset, the image file, grid file, geometric model, and, if the effects terrain are to be removed, a DEM must be opened.

4.3.3.6.3 Resample Level1R Imagery

Loop on each band of each SCA for resampling.

1. Get resampling grid for the band and SCA to be processed.
2. Build SCA-trimming table.
3. Read one band of imagery for one SCA.
   3.1. Initialize jitter correction parameters for this band (jitter_scale = 1 for TIRS)
4. Loop on output line/samples
   4.1. Check to see if output line/sample is within SCA-trimming bounds.
   ```
   if  output sample > LUT[ 2 * output line ] &&
           output sample < LUT[ 2 * output line + 1 ]  then proceed
   else       output pixel = fill
   ```
   4.2. If image is terrain corrected, calculate elevation dependent input line/sample location.
      4.2.1) Get elevation for output pixel location X/Y location from DEM (elevation).
      4.2.2) Map the output line/sample back into input space using the grid and the function 3d_ols2ils.
   4.3. If image is not terrain corrected calculate zero elevation (ellipsoid surface) input
   line/sample location.
      4.3.1) Set elevation to zero
      4.3.2) Map the output line/sample back into input space using the grid and the function ols2ils.
   4.4. Calculate actual input sample location; for sample location (int)input sample calculated from either 4.2 or 4.3:
      4.4.1) Calculate detector offset parallax scale.
      Scale = (int) floor(detector along-track offset + 0.5) (in geometric model).

4.4.2) Calculate sample detector parallax offset

$\Delta sample\_oe = (d_0 + elevation * d_1) * scale$

Note that $(d_0 + elevation * d_1)$ is the parallax (in pixels) per pixel of along-track offset from the nominal detector location.

Where:

$d_{0,1}$ = detector sample parallax coefficients stored in the grid

4.4.3) Get sample fractional offset

fractional sample offset =

detector across-track offset (in geometric model)

4.4.4) Calculate sample jitter adjustment

4.4.4.1) Calculate the index into the jitter table for the current image line

jit_index = (int)(jitter_scale*(input line – pixel column fill (defined below)))

Make sure jitter index is within the range of the jitter table. Set to the min or max value (whichever is closest) if it is outside the range.

4.4.4.2) Calculate the fractional jitter table index

Δjit_index = jitter_scale * input line – floor(jitter_scale * input line)

4.4.4.3) Calculate simple sample jitter adjustment

```
samp_jitter0 = samp_sens[0] * jitter_table[jit_index].roll
             + samp_sens[1] * jitter_table[jit_index].pitch
             + samp_sens[2] * jitter_table[jit_index].yaw
samp_jitter1 = samp_sens[0] * jitter_table[jit_index+1].roll
             + samp_sens[1] * jitter_table[jit_index+1].pitch
             + samp_sens[2] * jitter_table[jit_index+1].yaw
samp_jitter = samp_jitter0 * (1-Δjit_index) + samp_jitter1*Δjit_index
```

Where:

samp_sens[0] is the sample direction jitter roll sensitivity,
samp_sens[1] is the sample direction jitter pitch sensitivity,
samp_sens[2] is the sample direction jitter yaw sensitivity,
for the current grid cell, from the TIRS grid.
jitter_table[n] is the jitter table roll-pitch-yaw vector for row n,
from the TIRS model.

4.4.4.4) Refine the sample jitter to compensate for line jitter

```
line_jitter0 = line_sens[0] * jitter_table[jit_index].roll
             + line_sens[1] * jitter_table[jit_index].pitch
             + line_sens[2] * jitter_table[jit_index].yaw
line_jitter1 = line_sens[0] * jitter_table[jit_index+1].roll
             + line_sens[1] * jitter_table[jit_index+1].pitch
             + line_sens[2] * jitter_table[jit_index+1].yaw
line_jitter = line_jitter0 * (1-Δjit_index) + line_jitter1*Δjit_index
```

Where:

line_sens[0] is the line direction jitter roll sensitivity,
line_sens[1] is the line direction jitter pitch sensitivity,
line_sens[2] is the line direction jitter yaw sensitivity,

for the current grid cell, from the TIRS grid.
This is the error in the line coordinate used above, due to line jitter.
samp_rate =
samp_sens[0]*(jitter_table[jit_index+1].roll-jitter_table[jit_index].roll)
+ samp_sens[1]*(jitter_table[jit_index+1].pitch-
jitter_table[jit_index].pitch)
+ samp_sens[2]*(jitter_table[jit_index+1].yaw-jitter_table[jit_index].yaw)
This is the rate of change of sample jitter with line coordinate.
samp_jitter += line_jitter*samp_rate
This is the sample jitter correction adjusted for the effects of line jitter.

4.4.5) actual input sample = input sample - Δsample_oe – samp_jitter - fractional sample offset. These corrections are subtracted rather than added because what we are doing here is, rather than adjusting the input space interpolation location, computing the apparent location of the detector to the left of the interpolation location to make sure we have the correct range of samples to feed the interpolation logic. If the above adjustments lead to the "actual input sample" being greater than (to the right of) the original input sample location, then we move our sample range one more sample to the left. We perform a similar calculation on the detector to the right of the input space interpolation location to make sure that we do not have to shift one more sample in that direction. See also the note in Section 4.6.2 below.

4.5. Create fractional pixel shift for current input location:
Δline = input line - (int) input line
Δsample = input sample - (int) input sample

4.6. Create aligned samples for Akima resampling by applying cubic convolution weights in line direction.

4.6.1. Loop on actual input sample location:
For hybrid sample = (int) actual input sample - 2 to (int) actual input sample + 3. One extra hybrid sample created to left and right of minimum number of samples needed for Akima interpolation)
For NN resampling, only the two closest hybrid sample locations are calculated, that for (int) actual input sample and (int) actual input sample + 1.

4.6.1.1. Calculate line and hybrid sample detector offset parallax scale
scale = (int) floor(detector along-track offset + 0.5) (in geometric model). .

4.6.1.2. Calculate detector offset, parallax, and jitter correction for hybrid detector.

4.6.1.2.1. Detector offset and parallax corrections.
Δline_oe = (c_0 + elevation * c_1) * scale + pixel column fill - nominal detector fill - at_offset[hybrid sample]
Δsample_oe = (d_0 + elevation * d_1) * scale
Where:
$c_{0,1}$ = detector line parallax coefficients stored in the grid

$d_{0,1}$ = detector sample parallax coefficients stored in the grid. Note that (c_0 + elevation * c_1) is the along-track parallax (in pixels) per pixel of along-track offset from the nominal detector location and (d_0 + elevation * d_1) is the across-track parallax (in pixels) per pixel of along-track offset from the nominal detector location.

4.6.1.2.2. Jitter correction

The sample jitter correction is calculated as described in Section 4.4.4 above. The line jitter correction is calculated as follows:

jit_index = (int)(jitter_scale*(input line – pixel column fill))

□jit_index = jitter_scale * input line – floor(jitter_scale * input line)

line_jitter0 = line_sens[0] * jitter_table[jit_index].roll
+ line_sens[1] * jitter_table[jit_index].pitch
+ line_sens[2] * jitter_table[jit_index].yaw

line_jitter1 = line_sens[0] * jitter_table[jit_index+1].roll
+ line_sens[1] * jitter_table[jit_index+1].pitch
+ line_sens[2] * jitter_table[jit_index+1].yaw

line_jitter = line_jitter0 * (1-□jit_index) + line_jitter1*□jit_index

Where:

line_sens[0] is the line direction jitter roll sensitivity,
line_sens[1] is the line direction jitter pitch sensitivity,
line_sens[2] is the line direction jitter yaw sensitivity,
for the current grid cell, from the TIRS grid.

This is the error in the line coordinate due to jitter.

line_rate =
line_sens[0]*(jitter_table[jit_index+1].roll-jitter_table[jit_index].roll)
+ line_sens[1]*(jitter_table[jit_index+1].pitch-jitter_table[jit_index].pitch)
+ line_sens[2]*(jitter_table[jit_index+1].yaw-jitter_table[jit_index].yaw)

This is the rate of change of line jitter with line coordinate.

line_jitter += line_jitter*line_rate

This is the line jitter correction adjusted for the second order effects of line jitter. Note the similarity to the sample correction described in 4.4.4.4.

4.6.1.3. Calculate new hybrid line location.

hybrid line = (int)floor(input line + Δline_oe + line_jitter).

Note that in this case we add the corrections since we are adjusting the interpolation location.

4.6.1.4. Calculate new fractional hybrid line location.

Δhybrid line = input line + Δline_oe + line_jitter – hybrid line

If |Δhybrid line| > 1 then the integer line index must be adjusted and Δhybrid line brought back into the -1 < Δhybrid line < 1 range .

4.6.1.5. For CC resampling, apply cubic convolution in line direction to hybrid sample line DNs.

4.6.1.5.1. Calculate cubic convolution weights.

$$w_{n+2} = \sum_{n=-1}^{2} f(n - \Delta\text{hybridline})$$

Where *f* is equal to cubic convolution function.

4.6.1.5.2. Apply cubic convolution weights to L1R DNs.

hybrid line DN = $w_0 * h_0 + w_1 * h_1 + w_2 * h_2 + w_3 * h_3$

Where:

w_0, w_1, w_2, w_3 = Cubic convolution weights for Δhybrid line.

h_0 = DN from L1R for hybrid sample, input line location - 1

h_1 = DN from L1R for hybrid sample, input line location

h_2 = DN from L1R for hybrid sample, input line location + 1

h_3 = DN from L1R for hybrid sample, input line location + 2

For NN resampling, use the fractional hybrid line location to select the closest integer line number, and extract the corresponding pixel value as the hybrid line DN for the current hybrid sample.

4.6.2. Calculate the apparent Akima pixel location for the current hybrid sample.

Akima pixel location x_i =

hybrid sample location - Δsample_oe - across-track detector offset (in geometric model) – samp_jitter (computed per Section 4.6.1.2.2 above).

Note that in this case the across-track terrain parallax and sample jitter effects are subtracted instead of added. This is because we are adjusting the apparent detector location relative to the output pixel interpolation point instead of adjusting the output pixel interpolation location itself. We must do it this way in the sample direction because the adjustments are different for each detector. As for the across-track offset term, which is also unique for each detector, the detector offset corrections are designed to be applied as line-of-sight corrections in the instrument coordinate system. As such, the along-track offset is a +X LOS correction and the across-track offset is a +Y LOS correction. The instrument +X axis is in the +line direction but the +Y axis is in the –sample direction, so this correction is also subtracted from the apparent detector location.

4.7. For CC resampling, calculate output DN using Akima interpolation and hybrid line/sample information from 4.6.1 and 4.6.2.

4.7.1. Calculate Akima weights according to pixel locations from 4.6.2.

$$m_0 = \frac{\mathrm{DN}_1 - \mathrm{DN}_0}{x_1 - x_0}$$

$$m_1 = \frac{\mathrm{DN}_2 - \mathrm{DN}_1}{x_2 - x_1}$$

$$m_2 = \frac{\mathrm{DN}_3 - \mathrm{DN}_2}{x_3 - x_2}$$

$$m_3 = \frac{\mathrm{DN}_4 - \mathrm{DN}_3}{x_4 - x_3}$$

$$m_4 = \frac{\mathrm{DN}_5 - \mathrm{DN}_4}{x_5 - x_4}$$

$$ak_0 = \mathrm{DN}_2$$

$$ak_1 = \frac{|m_3 - m_2| * m_1 + |m_1 - m_0| * m_2}{|m_3 - m_2| + |m_1 - m_0|}$$

$$ak_2 = \frac{(3.0 * m_2 - 2.0 * ak_1 - \frac{|m_4 - m_3| * m_2 + |m_2 - m_1| * m_3}{|m_4 - m_3| + |m_2 - m_1|}}{x_3 - x_2}$$

$$ak_3 = \frac{ak_1 + \frac{|m_4 - m_3| * m_2 + |m_2 - m_1| * m_3}{|m_4 - m_3| + |m_2 - m_1|} - 2.0 * m_2}{(x_3 - x_2)^2}$$

Where:
DN_n = hybrid DNs calculated from cubic convolution, step 4.6.1.
x_n = Akima locations calculated in step 4.6.2.
ak_n = Akima weights
m_n = Akima slopes

4.7.2. Calculate output pixel DN using Akima A method.

$$\mathrm{outputDN} = ak_0 + ak_1 * ds + ak_3 * ds^2 + ak_4 * ds^3$$

Where

$$ds = (\Delta\mathrm{sample} + x_2)$$

The output sample point is located between hybrid samples x_2 and x_3 where x_n is from n=0…5.

For NN resampling, test the hybrid Akima locations for the extracted hybrid samples to decide which is closest to the desired output location. Select the closest hybrid sample value as the output DN.

4.8. Write output DN to image file.

5. Write out data descriptor record for image file. The baseline contents of the metadata record are shown in Table 4-39. All fields present in the table refer to the imagery associated with the DDR unless otherwise specified. Note that the scene roll angle is a new field added for off-nadir acquisitions. It would be

computed from the TIRS LOS model by interpolating the roll angle from the "original" attitude data sequence at the time corresponding to the precision model reference time t_ref. This would be done using the logic described in the Find Attitude sub-algorithm in the TIRS LOS Projection Algorithm, except operating on the "original" rather than the "corrected" attitude data sequence. The logic for using the "original" data is so that this scene roll value will not change due to LOS model correction. The sign convention on the roll angle is such that a positive roll angle would correspond to a positive orbital Y coordinate, which is looking to starboard.

4.3.3.6.4 Combining SCAs into one output file.

For an SCA combined output image the overlap region between SCAs can be handled by averaging the pixels between SCAs.

Input and Output File Details

Output is an L1G image file formatted according to LSDS 1822 Landsat 8 (L8) Level 1 (L1) Data Format Control Book (DFCB). The output is a HDF5 file. The metadata associated with the output file is listed below. This table follows the metadata fields in the L8 L1 DFCB. The metadata is split up into a file metadata and band metadata. For further information on this format see the L8 L1 DFCB. Not all fields within the prototype metadata fields are filled in with valid data.

Field	Description	Type
Spacecraft source	*Spacecraft associated with data record*	*char[32]*
Instrument source	*Imaging instrument (TIRS)*	*char[32]*
WRS path	WRS path number	integer
WRS row	WRS row number	integer
Capture direction	*Ascending or descending pass*	*char[64]*
Capture date	*Date imagery was acquired by instrument*	*char[11]*
Capture time	*Time of day imagery was acquired by instrument*	*char[8]*
Scene roll angle	*Roll angle (in degrees) at the scene center*	*float*
Correction type	*Raw, L1R, L1G, L1Gt, L1T*	*char[8]*
Acquisition type	*Earth, lunar, stellar*	*char[8]*
Projection Code	Map projection code	integer
Zone code	UTM zone code	integer
Datum code	Datum code for map projection	char[16]
Spheroid code	Earth model for map projection	integer
Projection units	Distance units	char[8]
Projection coefficients	Parameters needed by coordinate transformation package.	float[16]
Resampling Type	Resampling method (CC, NN)	char[4]
Software Version	Software version used to create image	char[11]
Sun Azimuth	Sun Azimuth at scene center	float
Sun Elevation	Sun Elevation at scene center	float
Sun Angles Valid	1 if sun data present 0 if not	char
Sun Angle Correction	None, scene center, per-pixel	char[32]
Earth Sun Distance	Earth sun distance	float
For each band:		

Field	Description	Type
Band Number	Landsat 8/9 band designation for current record	integer
Number lines	Number of lines present in data file	integer
Number samples	Number of samples present in data file	integer
Data Type	Data type of imagery	integer
Maximum Pixel	Maximum DN value in data	float
Minimum Pixel	Minimum DN value in data	float
Maximum Radiance	Maximum radiance	float
Minimum Radiance	Minimum radiance	float
Upper left projection coordinate	Upper-left y (latitude/northing) and x (longitude/easting) coordinate	float[2]
Upper right projection coordinate	Upper-right y (latitude/northing) and x (longitude/easting) coordinate	float[2]
Lower right projection coordinate	Lower-right y (latitude/northing) and x (longitude/easting) coordinate	float[2]
Lower left projection coordinate	Lower-left y (latitude/northing) and x (longitude/easting) coordinate	float[2]
Projection distance y	Pixel size for y map coordinate	float
Projection distance x	Pixel size for x map coordinate	float
Maximum Pixel Value	Maximum DN of pixels	float
Minimum Pixel Value	Minimum DN of pixels	float
Pixel range valid	Indicates valid max / min is listed	char
Maximum Radiance	Maximum radiance value	float
Minimum Radiance	Minimum radiance value	float
Spectral Radiance Scaling Offset	Offset to convert to radiance	float
Spectral Radiance Scaling Gain	Gain to convert to radiance	float
Radiance valid	Indicates valid radiance items are present	char
Instrument Source	Source of data	char[32]

Table 4-39. Metadata Contents

4.3.3.7 Notes

1. The bad detector replacement approach used by TIRS, in which detectors from the redundant row are swapped for bad detectors in the primary row, is similar to the detector select capability used by the OLI. TIRS will use the same detector offset approach, in which the along-track detector offsets are stored in the CPF with the whole pixel adjustment needed due to the detector selected and the small subpixel adjustment, capturing deviations from the nominal Legendre polynomial LOS model, that was present in the heritage ALI CPF detector offset field. The fractional detector offset is separated from the detector select offset at times during processing.
2. The TIRS LOS model will not specifically address the problem of multiple terrain intersections. A terrain occlusion mask (Terrain Occlusion Algorithm 4.2.5) will be generated to identify obstructed OLI pixels, and the TIRS grid structure should be compatible with the terrain occlusion algorithm but that algorithm will not be modified to accommodate TIRS.

3. The baseline (non-threaded) resampler implementation generates combined-SCA images by simply overwriting pixels present in multiple SCAs. Thus, the output image will contain pixel values from the highest-numbered SCA that views each pixel. The more sophisticated threaded resampler explicitly merges overlapping pixels, taking the average of pixels seen by two SCAs. Either method is acceptable but the latter approach is preferred.

4.3.4 TIRS Band-to-Band Calibration Algorithm

4.3.4.1 Background/Introduction

The TIRS Band-to-Band calibration (B2BCal) algorithm estimates improved values for band placement within each Sensor Chip Assembly (SCA) of the TIRS instrument. Adjustments are made relative to the primary detector row 10.8 micrometer band, or in other words, the 10.8 micrometer band serves as the reference for all other bands. The baseline algorithm calibrates only the TIRS primary row 12.0 micrometer band relative to the primary row 10.8 micrometer band. Some features are retained from the OLI band calibration algorithm that would facilitate adding the capability to calibrate the TIRS redundant rows in the future.

The B2B calibration takes the TIRS Band Accuracy Assessment residuals file, which represents displacements with respect to the product output projection space, maps the residuals back into displacements with respect to the focal plane and then performs a least squares (LSQ) fit between the focal plane residuals to determine updates to the TIRS band Legendre LOS polynomial coefficients. The least squares fit results represent updates needed to adjust the existing Legendre LOS coefficients. These updates can be used to produce new Legendre LOS coefficients for the CPF.

TIRS band alignment calibration algorithm would be applied to the output of the TIRS Band Registration Accuracy Assessment algorithm derived from SCA-separated TIRS images.

4.3.4.2 Dependencies

The TIRS B2B calibration algorithm assumes that a cloud free nadir viewing L1TP image has been generated and the resampled DEM used to create the L1TP is available. The TIRS LOS Model Creation and TIRS LOS Projection/Gridding algorithms (Sections 4.3.1 and 4.3.2) for the L1TP will be assumed to have been executed and the corresponding output files available. The L1TP image needs to be in SCA-separated format and either in a SOM or UTM path-oriented projection. The digital orthophoto quadrangle (DOQ) control and a digital elevation model (DEM) need to be used in generating the L1TP. The accuracy of the precision solution should have post-fit residuals below the recommended threshold, the solution should have used an adequate number of control points, and the distribution of the control should be well distributed throughout the imagery. The TIRS Band Registration Accuracy Assessment (BRAA), or Band Characterization (B2BChar), algorithm will assumed to have been run on the L1TP image successfully producing a B2B residuals file.

4.3.4.3 Inputs

The B2B calibration algorithm uses the inputs listed in the following table. Note that some of these "inputs" are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs	Algorithm Inputs
TIRS resampling grid	ODL
DEM	ODL
TIRS CPF file name	ODL
Along-track IFOV	CPF/LOS-model (See note #8)
Minimum points	ODL
Number of Legendre Coefficients	ODL (See note #6)
TIRS Line-of-Sight model	ODL
B2B residuals file	ODL
Band calibration report file	ODL
Trend flag	ODL
Flag for CPF group creation (see note #3)	ODL
Flag for individual tie-point listing	ODL
CPF effective dates (begin and end)	ODL
Work Order ID (for trending)	ODL

4.3.4.4 Outputs

B2B calibration report file (See note #1)
Legendre LOS CPF group
B2B calibration trending
Geometric Characterization ID
Work Order ID
WRS Path/Row
B2B calibration post and pre fit residuals
New SCA line-of-sight parameters

4.3.4.5 Options

Trending on/off switch

4.3.4.6 Procedure

Band calibration uses the residuals measured during the TIRS Band Registration Accuracy Assessment Algorithm (see Section 4.3.6) to determine updates to the Legendre LOS coefficients (see TIRS Line-of-Sight Model Creation Algorithm, Section 4.3.1). The band calibration process involves taking the residuals from Band Registration Accuracy Assessment, measured in output space, mapping them into input space angular deltas in terms of along- and across-track LOS angles and performing a least squares fit of the input space LOS angle deltas to a set of 3rd order Legendre polynomial correction coefficients. The correction polynomials calculated represent updates to the original LOS Legendre polynomial coefficients. New Legendre LOS coefficients can be found by combining the correction coefficients with the original coefficients.

Due to the differences in viewing geometry between bands within a SCA, along with the differences in viewing geometry between SCAs, the effects due to relief displacement must be taken into account during band calibration. To account for relief displacement during B2B calibration a DEM is required. The resampling grid and LOS model is also required during B2B calibration. The resampling grid, the corresponding detector's IFOV, and the LOS model's Legendre coefficients are used to map the residuals from output space to angular differences in input space.

A least squares fit is done on all requested bands and SCAs using the band-to-band tie point measurements from all band-pair combinations for a single SCA at a time. Requested bands and SCAs to process are based on the bands and SCAs present within the TIRS Band Registration Accuracy Assessment residuals file.

4.3.4.6.1 Stage 1 - Data input

The data input stage involves loading the information required to perform the band calibration. Input file names are needed for: geometric LOS resampling grid, LOS model, Band Registration Accuracy Assessment results (B2B residuals file), output band calibration report file name, and the L1TP DEM file name. Further input parameters are the effective begin and end dates of the new Legendre LOSs calculated, trending flag, CPF group creation flag, and individual tie-point reporting. Once the file names for the input data needed are retrieved the files can be opened and read.

Get ODL Parameters
Reads the parameters from the input ODL parameter file. This process was modified from the ALIAS heritage version to handle new inputs: minimum points, flag for CPF group creation, CPF effective dates, and flag for reporting individual tie-point results. The minimum points variable ensures that the normal matrix contains a minimum number along its diagonal to zero out any omitted bands. Rather than being removed from the solution, the offsets for omitted bands are set to zero with a weight equal to the minimum number of points.

Read Band-to-Band Residual File
Reads band accuracy assessment residuals file.

Read DEM
Read DEM file into IMAGE data structure.

Read TIRS LOS Model
Read TIRS geometric/LOS model.

Read TIRS LOS Geometric Grid
Read TIRS LOS resampling grid.

4.3.4.6.2 Stage 2 - Setup Least Squares Matrices and Solve

For each input SCA, every residual for each input band combination that is not an outlier is mapped back to TIRS input space. These input space mappings are single value adjustments needed for each point to align the LOS, associated with the focal plane, between the bands of the combination. This mapping procedure is described in more detail below. Once all of these residuals are mapped back to the focal plane and stored within the least-squares (LSQ) matrices new LOSs can be calculated.

The matrices defining calibration the process takes the following form:

$$[A][coeff] = [Y]$$

The matrices [*A*] and [*Y*] shown above correspond to one tie point measurement. The matrix [*coeff*] are the unknown adjustments to the Legendre LOS coefficients, the matrix [*A*] contain the Legendre coefficient multipliers for the band combination corresponding to that one measurement, and the [*Y*] matrix contains the input space residuals for that one measurement. For one measurement the matrices have the following dimensions:
[coeff] = (2 * Number of Legendre (4) * Number of bands (2)) x 1 = M x 1
[A] = 2 x (2 * Number of Legendre (4) * Number of bands (2)) = 2 x M
[Y] = 2 x 1

$$[coeff] = \begin{bmatrix} a_{b1,0} \\ a_{b1,1} \\ a_{b1,2} \\ a_{b1,3} \\ b_{b1,0} \\ b_{b1,1} \\ b_{b1,2} \\ b_{b1,3} \\ a_{b2,0} \\ a_{b2,1} \\ a_{b2,2} \\ a_{b2,3} \\ b_{b2,0} \\ b_{b2,1} \\ b_{b2,2} \\ b_{b2,3} \end{bmatrix}$$

Where:
$a_{bi,j}$ = Legendre coefficient j for line direction (along-track) for band i
$b_{bi,j}$ = Legendre coefficient j for sample direction (across-track) for band i

j = 0, 1, 2, 3 or the Number of Legendre coefficients to solve.
i = 1, 2 (Number of TIRS bands)

A 2x1 matrix pertaining to one residual measurement can be defined as:

$$[Y] = \begin{bmatrix} \Delta line \\ \Delta sample \end{bmatrix}$$

Where:
Δ*line* = input space residual in line direction (angular)
Δ*sample* = input space residual in sample direction (angular)

The TIRS input space residuals are calculated by finding the nominal (search) LOS in input space and the measured (search + measured offset) LOS in input space. These LOSs are found by mapping the output space line and sample locations to input space line and sample locations using the TIRS LOS projection grid (See TIRS Resampling Algorithm) and then using the TIRS LOS model (see TIRS Line-of-Sight Model Creation Algorithm) to convert the input space locations to LOSs. These input space nominal and measured locations are also used to construct the Legendre coefficient multipliers.

The design matrix [*A*] for one residual measurement is then:

$$[A_{nik}][coeff] = [Y_n]$$

$$[A] = \begin{bmatrix} -rl_{n,1,0} & -rl_{n,1,1} & -rl_{n,1,2} & -rl_{n,1,3} & 0 & 0 & 0 & 0 & sl_{n,2,0} & sl_{n,2,1} & sl_{n,2,2} & sl_{n,2,3} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -rl_{n,1,0} & -rl_{n,1,1} & -rl_{n,1,2} & -rl_{n,1,3} & 0 & 0 & 0 & 0 & sl_{n,2,0} & sl_{n,2,1} & sl_{n,2,2} & sl_{n,2,3} \end{bmatrix}$$

Where:
$rl_{n,1,j}$ = reference band (1)1 Legendre polynomial
$sl_{n,2,j}$ = search band (2) Legendre polynomial
j = 0, 1, 2, 3 or the Number of Legendre coefficients to solve
n = tie-point number

These matrices define one observation. A sequence of observations can be summed to define the normal equations for a set of coefficients that can be used to update the TIRS LOS Legendre coefficients:

$$[N] = \sum A_{nik}^T W_{ik}^{-1} A_{nik}$$
$$[L] = \sum A_{nik}^T W_{ik}^{-1} Y_{nik}$$

Where [*N*] and [*L*] are summed over all n. Note that for TIRS there is only one band combination (the reference 10.8 micrometer band and the search 12.0 micrometer band). *W* is a weight matrix that is currently set to the same weight for all observations.

Since all of the tie point observations involve band differences, the solution lacks an absolute reference. To stabilize the solution a constraint observation is added to provide

such a reference. This additional observation is applied to the 10.8 micrometer reference band as an offset of zero for each direction (line and sample). This fixes the reference band adjustment at zero and forces the search band to be registered to it.

$$[A_{00}] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$[Y_{00}] = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Where the 10.8 micrometer reference band is stored in the first eight columns of the [A] observation matrix.

The solution for a new set of Legendre coefficients is then:

$$[coeff] = [N]^{-1}[L]$$

Band Calibration Processing Steps

Note: Array indexes are zero-relative. Band numbers are 10.8 μm = 1, 12.0 μm = 2,
nLeg = Number of Legendre update coefficients to solve (1, 2, 3, 4 valid options).
Matrix indexes are zero relative

1. Initialize parameters

$$[W] = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix}$$

Where $\sigma^2 = 16$, an approximate measurement variance for the tie point observations.

Set up Least Squares Matrices

2. For each SCA to process

Initialize pre-fit statistics variables

pre-fit sum line = 0
pre-fit sum sample = 0
pre-fit sum line^2 = 0
pre-fit sum sample^2 = 0

Initialize LSQ matrices to zero

[N] = [0]
[L] = [0]

2.1. For the single TIRS band combination

rband = 10.8 µm reference band
sband = 12.0 µm search band

2.1.1. For each tie-point

Calculate reference line, sample location and search adjusted line, sample location.

rline = tie-point reference line location
rsamp = tie-point reference sample location
sline = tie-point search line location + line offset measured
ssamp = tie-point search sample location + sample offset measured

Note: sline, ssamp is the adjusted (or true) search location. rline, rsamp, sline, ssamp are output space pixel locations.

2.1.2. Set rband and sband to zero-relative (needed for matrix operations

rband = rband - 1
sband = sband - 1

Map residuals to input space (focal plane space).

2.1.3. Find elevation for reference and sample locations

relev = elevation at rline,rsamp
selev = elevation at sline,ssamp

2.1.4. Map rline,rsamp and sline,ssamp to input space using 3d_ols2ils and the search band TIRS LOS/resampling grid.

(riline,risamp) = 3d_ols2ils(search_grid, relev, rline, rsamp)
(siline,sisamp) = 3d_ols2ils(search_grid, selev, sline, ssamp)

Where

riline, risamp is the input space location of reference tie-point location.
siline, sisamp is the input space location of adjusted search tie-point location.
search_grid is the TIRS LOS/resampling grid for the search band.

Note: Search band grid is used for mapping both the adjusted search (siline,sisamp) and the reference locations.

2.1.5. Calculate Legendre normalized detector location

$$\text{rnorm} = \frac{2.0 * \text{risamp}}{\text{number detectors in SCA} - 1} - 1$$

$$\text{snorm} = \frac{2.0 * \text{sisamp}}{\text{number detectors in SCA} - 1} - 1$$

rnorm = normalized reference detector
snorm = normalized adjusted search detector

2.1.6. Calculate reference and search along- and across-track LOS.

$$nom\_sear\_x = coef\_x_{s,0} + coef\_x_{s,1} * rnorm + coef\_x_{s,2} * (3 * rnorm^2 - 1)/2$$
$$+ coef\_x_{s,3} * rnorm * (5 * rnorm^2 - 3)/2$$

$$nom\_sear\_y = coef\_y_{s,0} + coef\_y_{s,1} * rnorm + coef\_y_{s,2} * (3 * rnorm^2 - 1)/2$$
$$+ coef\_y_{s,3} * rnorm * (5 * rnorm^2 - 3)/2$$

$$sear\_x = coef\_x_{s,0} + coef\_x_{s,1} * snorm + coef\_x_{s,2} * (3 * snorm^2 - 1)/2$$
$$+ coef\_x_{s,3} * snorm * (5 * snorm^2 - 3)/2$$

$$sear\_y = coef\_y_{s,0} + coef\_y_{s,1} * snorm + coef\_y_{s,2} * (3 * snorm^2 - 1)/2$$
$$+ coef\_y_{s,3} * snorm * (5 * snorm^2 - 3)/2$$

Where

ref_x, ref_y = along and across-track view angles
sear_x, sear_y = along and across-track view angles
coef_x$_{s,n}$ = search Legendre along-track coefficients
coef_y$_{s,n}$ = search Legendre across-track coefficients

2.1.7. Determine LOS vectors

sear_z = 1.0

$$m = \sqrt{sear\_x^2 + sear\_y^2 + sear\_z^2}$$

$$sear\_x = \frac{sear\_x}{m}$$

$$sear\_y = \frac{sear\_y}{m}$$

$$sear\_z = \frac{sear\_z}{m}$$

nom_sear_z = 1.0

$$m = \sqrt{nom\_sear\_x^2 + nom\_sear\_y^2 + nom\_sear\_z^2}$$

$$nom\_sear\_x = \frac{nom\_sear\_x}{m}$$

$$nom\_sear\_y = \frac{nom\_sear\_y}{m}$$

$$nom\_sear\_z = \frac{nom\_sear\_z}{m}$$

Determine effective line-of-sight instantaneous-field-of-view (IFOV)

2.1.7. Map input search pixel location, line and sample, to output space.

sline = search line location
ssamp = search sample location
elevation = elevation for location sline, ssamp

2.1.7.1. Calculate elevation planes bounding current elevation.

$$zplane = \frac{elevation}{grid\ z\ spacing} + grid\ zero\ plane$$

*elev0 = grid z spacing * (zplane – grid zero plane)*
elev1 = elev0 + grid z spacing

2.1.7.2. Calculate cell index, row and column, for search line and sample location and zplane.

row = sline / grid cell line spacing
column = ssamp / grid cell sample spacing
*cell index0 = nrows * ncols * zplane + row * ncols + column*

Where:

grid z spacing = elevation difference between two grid planes
ncols = number of grid cell columns
nrows = number of grid cell rows

2.1.7.3. Calculate output space line, sample location for input space search line, sample location and zplane.

$a_{0,1,2,3}$ = grid sample location forward mapping coefficients for cell index0
$b_{0,1,2,3}$ = grid line location forward mapping coefficients for cell index0

*lms = sline * ssamp*
*osamp0 = a_0 + a_1 * ssamp + a_2 * sline + a_3 *lms*
*oline0 = b_0 + b_1 * ssamp + b_2 * sline + b_3 * lms*

2.1.7.4. Calculate cell index, row and column, for search line and sample location and zplane +1.

*cell index1 = nrows * ncols * (zplane + 1.0) + row * ncols + column*

2.1.7.5. Calculate output space line, sample location for input space search line, sample location and zplane+1.

$a_{0,1,2,3}$ = grid sample location forward mapping coefficients for cell index1
$b_{0,1,2,3}$ = grid line location forward mapping coefficients for cell index1
*lms = sline * ssamp*
$osamp1 = a_0 + a_1 * ssamp + a_2 * sline + a_3 * lms$
$oline1 = b_0 + b_1 * ssamp + b_2 * sline + b_3 * lms$

2.1.7.6. Calculate output space line, sample location for input space search line, sample location, and elevation.

w0 = (elev1 – elevation) / (elev1 – elev2)
w1 = (elevation – elev0) / (elev1 – elev2)
$osamp_n = osamp0 * w0 + osamp1 * w1$
$oline_n = oline0 * w0 + oline1 * w1$

2.1.7.7. Map input location ssamp, sline+1.0 to output space $osamp_{n+1}$,$oline_{n+1}$ (repeat steps 2.1.7.1 to 2.1.7.6 for input location ssamp,sline+1)

2.1.7.8. Determine change in output space between input locations (ssamp, sline) and (ssamp, sline+1.0)

dline = $oline_n$ – $oline_{n+1}$
dsamp = $osamp_n$ = $osamp_{n+1}$

$$\text{distance} = \sqrt{\text{dline*dline} + \text{dsamp*dsamp}}$$

If Earth acquisition

2.1.7.9.1. Calculate output latitude and longitude for search line and sample (see Forward Model section of the TIRS LOS Projection/Grid Generation Algorithm Section 4.3.2).

2.1.7.9.2. Calculate time for current search line and sample (see Forward Model section of the TIRS LOS Projection/Grid Generation Algorithm Section 4.3.2).

2.1.7.9.3. Calculate satellite position for current search line and sample time (see Forward Model section of the TIRS LOS Projection/Grid Generation Algorithm Section 4.3.2).

2.1.7.9.4. Calculate target vector (see Forward Model section of the TIRS LOS Projection/Grid Generation Algorithm Section 4.3.2).

LOS x coordinate = target x coordinate - satellite x coordinate
LOS y coordinate = target y coordinate - satellite y coordinate
LOS z coordinate = target z coordinate - satellite z coordinate

length = sqrt(LOS x * LOS x + LOS y * LOS y + LOS z * LOS z)

$IFOV_{along}$ = (output pixel size * distance) / length

If lunar acquisition

$IFOV_{along}$ = output pixel size * distance

2.1.7.9.5. Calculate the LOS errors

$$\Delta line = \frac{sear\_x}{sear\_z} - \frac{nom\_sear\_x}{nom\_sear\_z} + (siline - riline) * IFOV_{along}$$

$$\Delta samp = \frac{sear\_y}{sear\_z} - \frac{nom\_sear\_y}{nom\_sear\_z}$$

Create matrices need to sum with [N] and [L].

2.1.7.10. Calculate Legendre generating polynomial coefficients for search and reference:

sl_0 = 1.0
if(*nLeg* >= 2) sl_1 = snorm
if(*nLeg* >= 3) sl_2 = (3 * snorm^2 – 1)/2
if(nLeg == 4) sl_3 = snorm * (5 * snorm^2 – 3)/2
rl_0 = 1.0
if (*nLeg* >= 2) rl_1 = rnorm
if(*nLeg* >= 3) rl_2 = (3 * rnorm^2 – 1)/2
if(nLeg == 4) rl_3 = rnorm * (5 * rnorm^2 – 3)/2

Note: If the number of Legendre coefficients in the solution is less than 4 the corresponding sl_n and rl_n will be omitted.

2.1.7.11. Initialize [A] to zero and then set [A] indexes to sl_n and rl_n.

A[0][Number Legendre * sband + n] = sl_n
A[1][Number Legendre * sband + n] = sl_n
A[0][Number Legendre * rband + n] = -rl_n
A[1][Number Legendre * rband + n] = -rl_n
Where: n = 0 … *nLeg* -1
[A] = 0 elsewhere

2.1.7.12. Set [Y] according to input space deltas measured and sum pre-fit statistics,

Store deltas in [Y]
Y[0][0] = Δ*line*
Y[1][0] = Δ*samp*

2.1.7.13. Sum statistics

pre-fit sum line = pre-fit sum line + Δ*line*
pre-fit sum sample = pre-fit sum sample + Δ*sample*
pre-fit sum line^2 = pre-fit sum line^2 + Δ*line*^{2}
pre-fit sum sample^2 = pre-fit sum sample^2 + Δ*sample*^{2}

2.1.7.14. Create matrices to add to normal matrices

$[A_{tie\text{-}point}] = [A]^T[W][A]$
$[Y_{tie\text{-}point}] = [A]^T[W][Y]$

2.1.7.15. Sum N and L matrices
$[N] = [N] + [A_{tie\text{-}point}]$
$[L] = [L] + [Y_{tie\text{-}point}]$

2.1.8. Set minimum points for bands to omit from processing.
Eliminate observations for omitted band:
oband = band to omit - 1 (from earlier, bands are 1-relative)
$[N]_{g+n,i} = 0$
Where g = *nLeg* * 2 * oband
n = 0 … 2**nLeg* - 1
i = 0 … *nLeg* * 2 * Number of Bands - 1
$[N]_{i,g+n} = 0$
Where g = *nLeg* * 2 * oband
n = 0 … 2**nLeg* - 1
i = 0 … *nLeg* * 2 * Number of Bands - 1
$[N]_{g+n,g+n}$ = Minimum Points
Where g = *nLeg* * 2 * oband
n = 0 … 2**nLeg* - 1
$[L]_{g+n} = 0$
Where g = *nLeg* * 2 * oband
n = 0 … 2**nLeg* - 1

Determine least squares solution and new Legendre coefficients
3. Solve for delta Legendre coefficients

$$[\Delta coeff] = [N]^{-1}[L]$$

4. Calculate new Legendre coefficients based on deltas calculated in step 3.0

$$new\ along\ coefficients_{SCA,band,n} = previous\ along\ coefficients_{SCA,band,n} + \Delta coeff_{i+n}$$
$$new\ across\ coefficients_{SCA,band,n} = previous\ across\ coefficients_{SCA,band,n} + \Delta coeff_{i+nLeg+n}$$

SCA = 1,2,3
band = 1, 2i = 2 * *nLeg* * (band – 1)
n = 0 … *nLeg* - 1

Calculate Pre and Post fit Residuals
5. For each SCA calculate residuals

5.1. Initialize post-fit statistics variables
post-fit sum line = 0
post-fit sum sample = 0
post-fit sumsq line = 0
post-fit sumsq sample = 0

5.2. For the single band combination

5.2.1. Perform steps 2.1.7 - 2.1.9 from stage 3.

5.2.2. Calculate adjusted reference and search line/sample locations

riline' = $\Delta coeff_{sca,rband,0}$

if(*nLeg* >=2) riline' = riline' + rnorm * $\Delta coeff_{sca,rband,1}$

if(*nLeg* >= 3) riline' = riline' + (3 * rnorm^2 - 1)/2 * $\Delta coeff_{sca,rband,2}$

if(*nLeg* == 4) riline' = riline' + rnorm * (5 * rnorm^2 - 3)/2 * $\Delta coeff_{sca,rband,3}$

risamp' = $\Delta coeff_{sca,rband,0}$

if(*nLeg* >= 2) risamp' = risamp' + rnorm * $\Delta coeff_{sca,rband,1}$

if(*nLeg* >= 3) risamp' = risamp' + (3 * rnorm^2 - 1)/2 * $\Delta coeff_{sca,rband,2}$

if(*nLeg* == 4) risamp' = risamp' + rnorm * (5 * rnorm^2 - 3)/2 * $\Delta coeff_{sca,rband,3}$

siline' = $\Delta coeff_{sca,rband,0}$

if(*nLeg* >= 2) siline' = siline' + snorm * $\Delta coeff_{sca,sband,1}$

if(*nLeg* >= 3) siline' = siline' + (3 * snorm^2 - 1)/2 * $\Delta coeff_{sca,sband,2}$

if(*nLeg* == 4) siline' = siline' + snorm * (5 * snorm^2 - 3)/2 * $\Delta coeff_{sca,sband,3}$

sisamp' = $\Delta coeff_{sca,sband,0}$

if(*nLeg* >= 2) sisamp' = sisamp ' + snorm * $\Delta coeff_{sca,sband,1}$

if(*nLeg* >= 3) sisamp' = sisamp' + (3 * snorm^2 - 1)/2 * $\Delta coeff_{sca,sband,2}$

if(*nLeg* == 4) sisamp' = sisamp' + snorm * (5 * snorm^2 - 3)/2 * $\Delta coeff_{sca,sband,3}$

Where:

SCA, band, 0, 1, 2, 3 are the SCA, band number and coefficients for the updates to the Legendre polynomials. The Δ*coeff* added to the riline are the along-track updates the Δ*coeff* added to the risamp are the across-track updates.

rband = index to reference band coefficient

sband = index to search band coefficient

5.2.3. Calculate new post fit Δ*errors* by updating Δ*line* and Δ*sample* with Legendre updates

Δ*line*' = Δ*line* - riline' + siline'

Δ*samp*' = Δ*samp* - risamp' + sisamp'

Where:

Δ*line* and Δ*samp* are the same as those calculated in 2.1.10 from stage 3.

5.2.4. Sum post-fit variables

post-fit sum line = post-fit sum line + Δ*line*'

post-fit sum sample = post-fit sum sample + Δ*sample*'

post-fit sumsq line = post-fit sumsq line + Δ*line*'^{2}

post-fit sumsq sample = post-fit sumsq sample + Δ*sample*'^{2}

5.3. Calculate post and pre-fit statistics for both line and sample directions:

$$\text{mean} = \frac{\text{sum}}{\text{number of points}}$$

$$\text{scale} = \frac{1}{\text{number of points} * (\text{number of points} - 1)}$$

$$\text{standard deviation} = \sqrt{\frac{\text{sum squares}}{\text{number of points} - 1} - \text{sum} * \text{sum} * \text{scale}}$$

$$\text{rmse} = \sqrt{\frac{\text{sum squares}}{\text{number of points}}}$$

Where:

sum = pre/post sum line or pre/post sum sample
sum squares = pre/post sumsq line or pre/post sumsq sample
number of points = number of points used in LSQ fit

5.4. Create Band-to-Band Calibration output report (see Table 4-40).

5.4.1. Write report header information.

5.4.2. Write post and pre-fit statistics (per SCA) for line and sample direction.

5.4.3. Write individual tie-point statistics (if tie-point reporting flag = Yes).

6. If CPF group flag is set to yes write out ASCII file of CPF group with new Legendre coefficients.

4.3.4.7 Output files

The output report contains a standard header. This standard header is at the beginning of the file and contains the following:

1. Date and time the file was created
2. Spacecraft and instrument pertaining to measurements.
3. Pointing (roll) angle of spacecraft/instrument.
4. Acquisition type
5. Report type (band-to-band)
6. Work order ID of process (left blank if not applicable)
7. WRS path/row
8. Software version that produced report.
9. L0R image file name

The following items should be stored (trended) in the database with respect to the Band-to-Band Calibration algorithm:

All report header information:
Date and time

Spacecraft instrument source
Work order ID
WRS path/row
Software version
Off-nadir angle
L0Rp file name
Processing file name

The following processing parameters:

Bands processed
SCAs processed

The following report file information:

Number of points used per SCA
Computed Legendre along-track coefficient updates
Computed Legendre across-track coefficient updates
New Legendre along-track coefficients (updates + existing)
New Legendre across-track coefficients (updates + existing)
Post-fit mean, standard deviation, RMSE
Pre-fit mean, standard deviation, RMSE

See note #11.

Field	Description	Trend
Date and time	Date (day of week, month, day of month, year) and time of file creation.	Yes
Spacecraft and instrument source	LDCM and TIRS	Yes
Processing Center	EROS Data Center SVT	No
Work order ID	Work order ID associated with processing (blank if not applicable)	Yes
WRS path/row	WRS path and row (See note #4)	Yes
Software version	Software version used to create report	Yes
Off-nadir angle	Off-nadir pointing angle of processed image file (See note #5)	Yes
Acquisition Type	Earth viewing or Lunar	Yes
L0Rp image file	L0Rp image file name used to create L1T	Yes
Processed image file name	Name of L1T used to create report	Yes
Number of Legendre coefficients	Number of Legendre coefficients present	Yes
Heading for pre and post fit statistics	One line of ASCII text defining pre and post statistics	
For each SCA (along and across track directions)		
SCA number	SCA number associated with statistics	Yes
Pre fit statistics	Mean, RMSE, standard deviation, along and across track direction (in units of radians)	Yes
Post fit statistics	Mean, RMSE, standard deviation, along and across track direction (in units of radians)	Yes
For each SCA and band		
Along track solution	Legendre along track correction coefficients	Yes
Across track solution	Legendre across track correction coefficients	Yes
For each SCA and band		
Along track updates	Updated Legendre along track coefficients	Yes
Across track updates	Updated Legendre across track coefficients	Yes

Field	Description	Trend
For each tie-point of each SCA and Band to process	Output produced only if tie-point results flag is set to Yes.	
Point ID	Point identifier	No
SCA number	SCA number for band combination	No
Reference output line	Output tie-point location in line direction	No
Reference output sample	Output tie-point location in sample direction	No
Reference input line	Reference input tie-point location in line direction	No
Reference input sample	Reference input tie-point location in sample direction	No
Search input line	Search input tie-point location in line direction	No
Search input sample	Search input tie-point location in sample direction	No
Reference band	Reference band	No
Search Band	Search band	
Measured line offset	Output space offset in line direction (from Band Accuracy Assessment residuals file)	No
Measured sample offset	Output space offset in sample direction (from Band Accuracy Assessment residuals file)	No
Pre-fit line delta	Pre-fit input space line delta/offset (Δ*line*)	No
Pre-fit sample delta	Pre-fit input space sample delta/offset (Δ*sample*)	No
Post-fit line delta	Post-fit input space line delta/offset (Δ*line*')	No
Post-fit sample delta	Post-fit input space sample delta/offset (Δ*sample*')	No

Table 4-40. Band Calibration Report File

If the CPF group creation flag is set to yes an ASCII file containing the updated Legendre LOS should be generated. This file would contain the new Legendre LOS for each SCA for every band and would be formatted according to the CPF group that the Legendre LOS resides in. The file would also contain the file attributes CPF group with the effective dates for the LOS generated (See note #3). The SCAs and bands that were not updated should still be represented within the file; these values should be the same for post and pre calibration.

4.3.4.8 Notes

Some additional background assumptions and notes include the following:

1. The band calibration results currently contains the L1TP name, pre and post fit mean, root mean square error, and standard deviation for the along and across-track direction of each SCA, new Legendre LOS coefficients, and a new CPF Legendre LOS group parameters. The individual tie-point characteristic information and (pre and post-fit) residuals and should be added to the report file (see Table 4-40).
2. See "Using the LOS geometric resampling grid to map an output pixel location to an input pixel location" in the TIRS Resampling Algorithm for ols2ils functionality.
3. Table 4-41 contains the file attributes and LOS groups that should be populated with the corresponding TIRS fields when the CPF group creation flag is set to yes.

Parameter Groups	Parameter Name	Data Type	Description
GROUP: LOS_LEGENDRE	Along_LOS_Legendre_B***BB_NNN***_SCA***SS***	float32 array (4 values) for each band of each SCA	Legendre polynomial coefficients defining along-track viewing angle of band number ***BB***, band name ***NNN*** and SCA ***SS*** given in radians Valid format: for each term: SN.NNNNESN, where S = “+” or “-,” N = 0 to 9, and E = "E.”
GROUP: LOS_LEGENDRE	Across_LOS_Legendre_B***BB_NNN***_SCA***SS***	float32 array (4 values) for each band of each SCA	Legendre polynomial coefficients defining across-track viewing angle of band number ***BB***, band name ***NNN*** and SCA ***SS*** given in radians Valid format: for each term: SN.NNNNESN, where S = “+” or “-,” N = 0 to 9, and E = "E"

Table 4-41. LOS CPF Groups

The file name for the CPF group can follow the convention of:
Legendre_coefficients_<effective begin date>_<effective end date>.odl
Where:
effective begin date = YYYYMMDD
effective end date = YYYYMMDD
YYYY = Year
MM = Month of year
DD = Day of month

4. Any kind of "non-WRS" collect; lunar or off-nadir viewing at the poles should have 000/000 listed as the path/row.
5. Pointing angle for lunar acquisitions would be 0.0.
6. Currently it is not expected that any calibration will be done on anything other than the full range of Legendre coefficients (4), however, support for a range of 1-4 Legendre coefficients in the solution should remain in the system.
7. The normal operating procedure will be to calibrate all of the TIRS SCAs. Input that does not contain the full three TIRS SCAs is more of a toolkit and testing capability.
8. The IFOV was historically used for calculating the LOS errors of the measured residuals. The code was modified to calculate a dynamic IFOV, however the CPF static IFOV was left as an input to allow for comparisons between the historical and current dynamic methods.

4.3.5 TIRS Alignment Calibration Algorithm

4.3.5.1 Background/Introduction

The TIRS alignment calibration algorithm combines the functions of the OLI sensor alignment and focal plane alignment calibration algorithms. Using an OLI SWIR band image as a reference it compares an SCA-separated precision and terrain corrected (L1T) TIRS 10.8 micrometer band image (see note #1) with the OLI SWIR reference image. Each SCA in the TIRS L1T image is compared to the SCA-combined OLI

reference to measure both systematic full-scene TIRS-to-OLI misregistration and SCA-specific deviations from the scene-average registration. The measured deviations are used to estimate corrections to the TIRS-to-OLI alignment matrix and to the 10.8 micrometer band Legendre polynomial coefficients that model the nominal lines-of-sight for each SCA.

The algorithm is implemented in two steps: 1) a mensuration/setup step in which the separated-SCA L1T image is correlated with the OLI reference image to measure the within-SCA deviations, and; 2) a calibration update computation step in which the measured deviations are used to compute TIRS-to-OLI alignment corrections and TIRS line-of-sight model correction Legendre coefficients that adjust the original LOS model to minimize the residual image deviations. The calibration update step includes applying an outlier filter to the image measurements. Separating the algorithm into two distinct steps makes it possible to run the calibration update step multiple times, using different outlier filter thresholds, for example, without having to perform the time consuming image mensuration/correlation setup procedure more than once.

Results from individual calibration scenes are stored in the geometric trending database so that results from multiple scenes can be analyzed together when deciding whether and how to adjust the operational ACS-to-TIRS alignment and TIRS focal plane calibrations. If a 10.8 micrometer band focal plane calibration update is generated, the TIRS 12.0 micrometer spectral band would subsequently be re-registered to the 10.8 micrometer band using the TIRS band alignment calibration procedure.

The TIRS alignment calibration procedure is derived from the OLI focal plane calibration algorithm. The implementation should be very similar for the setup step, which measures the SCA-specific deviations relative to the reference image. The legendre step, which calculates the Legendre polynomial coefficient updates, will be enhanced to include the computation of the full-scene TIRS-to-OLI alignment update. Since the Legendre coefficients and alignment angles are not completely independent, some additional constraints are required to make the parameters separable. The default approach is to constrain the Legendre coefficients so that they cannot model roll, pitch, or yaw effects (more about this below). An alternate option is to constrain the alignment angles. This option makes the calibration solution mimic the OLI focal plane calibration procedure.

4.3.5.2 Dependencies

The TIRS alignment calibration algorithm assumes that the L1T process flow has created a substantially cloud-free SCA-separated (nadir-viewing) path-oriented L1T 10.8 micrometer band image, over a band registration calibration site, which has been registered to an OLI reference image either by using systematic LOS models for both the OLI and TIRS images, or by transferring the OLI-derived precision correction model to the TIRS LOS model. This occurs automatically in the combined OLI/TIRS LOS model. The SCA-separated TIRS L1T image will be framed to exactly match the SCA-combined OLI reference image. This algorithm also assumes that the CPF, TIRS LOS model, TIRS grid file, and DEM used to produce the TIRS L1T image, are available.

4.3.5.3 Inputs

The TIRS alignment calibration algorithm uses the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data). The second column shows which algorithm step (image mensuration or correction model computation) uses the input.

Algorithm Inputs	Processing Step
ODL File (implementation)	Both
CPF Name	Both
TIRS L1T Image File Name (see note #2)	Step 1
TIRS LOS Model File Name	Step 1
TIRS LOS Projection Grid File Name	Step 1
DEM File Name	Step 1
OLI Reference Image File Name	Step 1
Correlation Data File Name	Both
Report File Name	Step 2
Processing Parameters	Both
Number of Tie Points per Cell	Step 1
Outlier tolerance	Step 2
Constraint Type: 0 (default) = constrain Legendre coefficients (solve all parameters), 1 = constrain angles (solve Legendre only)	Step 2
Work order ID (for trending)	Step 2
WRS Path (for trending)	Step 2
WRS Row (for trending)	Step 2
Calibration effective dates for updated parameters	Step 2
Trending flag	Step 2
CPF	Both
Calibration effective dates	Step 2
ACS-to-OLI alignment matrix (3x3 orientation matrix)	Step 2
ACS-to-TIRS alignment matrix (3x3 orientation matrix)	Step 2
Algorithm Parameters (formerly system table parameters)	
Size of Correlation Window	Step 1
Peak Fit Method	Step 1
Min Correlation Strength	Step 1
Max Correlation Displacement	Step 1
Fill Threshold Fraction (max percent of window containing fill value)	Step 1
Tie point weight (in units of 1/microradians^2)	Step 2
Alignment constraint weight (in units of 1/microradians^2) (new)	Step 2
Fit order	Step 2
Post-fit RMSE Thresholds (trending metrics) (in units of microradians)	Step 2
TIRS Grid File (see LOS Projection Algorithm for details)	Step 1
Number of SCAs	Step 1
For each SCA:	Step 1
Grid cell size in lines/samples	Step 1

Algorithm Inputs	Processing Step
Number of lines/samples in grid	Step 1
Number of z-planes, zero z-plane index, z-plane spacing	Step 1
Array of grid input line/sample locations	Step 1
Array of output line/sample locations (per z-plane)	Step 1
Array of forward mapping coefficients	Step 1
Array of inverse mapping coefficients	Step 1
Rough mapping polynomial coefficients	Step 1
TIRS LOS Model File	Step 1
TIRS Along-Track IFOV (in radians)	Step 1
Number of SCAs	Step 1
Number of Bands	Step 1
Number of Detectors per SCA per Band	Step 1
Focal Plane Model Parameters (Legendre Coefficients) (in radians)	Step 2
ACS-to-TIRS Alignment Matrix (3x3 orientation matrix) (see note #5)	Step 2
TIRS L1T Image (separated SCA)	Step 1
Image corner coordinates	Step 1
Pixel size (in meters)	Step 1
Image size	Step 1
Search image pixel data (10.8 micrometer band)	Step 1
DEM	Step 1
DEM corner coordinates	Step 1
Pixel size (in meters)	Step 1
DEM size	Step 1
Elevation data	Step 1
OLI Reference Image	Step 1
Image corner coordinates	Step 1
Pixel size (in meters)	Step 1
Image size	Step 1
Reference image pixel data (SWIR1 or SWIR2 band)	Step 1
Correlation Data File (output of Step 1)	Step 2
Correlation results in TIRS input space pixels	Step 2
LOS errors in radians	Step 2
Correlation results in output space pixels	Step 2

4.3.5.4 Outputs

Step 1: TIRS Alignment Setup
Correlation Data File (temporary output passed to Update step)
Correlation results in output space pixels
Correlation results in TIRS input space pixels
LOS errors in radians
Step 2: TIRS Alignment Update (see Table 4-43 below)
Report File (see Table 4-43 below for details)
Standard report header
Acquisition date
Ref (OLI)/Search (TIRS) image names
Constraint Type (Legendre or Angle)

Original TIRS-to-OLI roll-pitch-yaw
Estimated TIRS-to-OLI roll-pitch-yaw correction
Updated TIRS-to-OLI roll-pitch-yaw
Original TIRS-to-OLI alignment matrix (3x3)
Updated TIRS-to-OLI alignment matrix (3x3)
Number of SCAs
For each SCA:
SCA Number
Old Along- and Across-track Legendre coefficients (NSCAx2x4)
Along- and Across-track Legendre error (fit) coefficients (NSCAx2x4)
New Along- and Across-track Legendre coefficients (NSCAx2x4)
Pre-fit along- and across-track offset statistics (mean, stddev, RMSE)
Post-fit along- and across-track residual statistics (mean, stddev, RMSE)
Confidence level used for outlier rejection
Legendre polynomial fit order (see note #3)
Number of tie points used for current SCA
CPF LOS_LEGENDRE and ATTITUDE_PARAMETERS Groups (separate ODL-format files) (see note #6)
Effective Dates (embedded in output file names)
New ACS-to-TIRS alignment matrix (3x3)
New Legendre polynomial coefficients (NSCAx2x4)
Measure Tie Point Data
For each point:
SCA Number
Grid Cell Column Number
Nominal Output Space Line
Nominal Output Space Sample
Measured LOS Error Delta Line (in pixels)
Measured LOS Error Delta Sample (in pixels)
Measured LOS Error Along-Track Delta Angle (in microradians)
Measured LOS Error Across-Track Delta Angle (in microradians)
Tie Point State (outlier) Flag
Along-Track Fit Residual (in microradians)
Across-Track Fit Residual (in microradians)
TIRS Alignment Trending Database (see Table 4-43 below for details)
Geometric Characterization ID
Work Order ID
WRS path/row
Acquisition date
Ref (OLI) image name
Constraint Type (Legendre or Angle)
Original TIRS-to-OLI roll-pitch-yaw
Estimated TIRS-to-OLI roll-pitch-yaw correction
Updated TIRS-to-OLI roll-pitch-yaw
Number of SCAs
For each SCA:

SCA Number
Old Along- and Across-track Legendre coefficients (NSCAx2x4)
Along- and Across-track Legendre error (fit) coefficients (NSCAx2x4)
New Along- and Across-track Legendre coefficients (NSCAx2x4)
Pre-fit along- and across-track offset statistics (mean, stddev, RMSE)
Post-fit along- and across-track residual statistics (mean, stddev, RMSE)
Confidence level used for outlier rejection
Number of tie points used for current SCA

4.3.5.5 Options

TIRS Alignment Calibration Trending On/Off Switch

4.3.5.6 Procedure

The TIRS Alignment Calibration Algorithm is used for on-orbit calibration of the TIRS-to-OLI instrument alignment as well as for the alignment of the lines-of-sight of the TIRS SCAs relative to each other. This calibration is necessary to meet the TIRS-to-OLI band registration, and the TIRS image registration, geodetic accuracy, and geometric accuracy requirements.

Procedure Overview

The TIRS alignment algorithm adjusts the overall TIRS field of view and each TIRS SCA to an OLI reference image. By simultaneously aligning the TIRS SCAs to a common reference, any measured inter-SCA misalignment is removed. Each TIRS SCA is correlated against a reference image created from an OLI SWIR band, acquired at approximately the same time. A new set of 3rd order Legendre LOS coefficients, representing updates or corrections to the original polynomials, are generated by fitting a set of coefficients to the measured LOS deviations. A set of roll-pitch-yaw angular adjustments are also computed to remove any alignment biases between the TIRS and OLI instruments.

Substantially cloud free scenes should be used for TIRS alignment calibration. The imagery should have ground control applied and terrain displacements removed, i.e., the imagery should be a terrain corrected (L1T) dataset. Both the OLI SWIR and TIRS images should be path oriented and resampled to 30m output pixel size.

4.3.5.6.1 Stage 1: Setup – Correlate L1T Image with OLI Reference

An array of test points is generated for each TIRS SCA based upon the number of points per grid cell specified in the input parameters. The TIRS LOS projection grid is used to generate the test point array by spacing the test points at regular intervals in TIRS input space, and then computing the corresponding output space coordinates for each. Constructing the test point array in the TIRS input space ensures that the test points fall within the active area of each TIRS SCA.

Image windows extracted from the L1T image at the test point locations are correlated with corresponding windows extracted from the reference OLI SWIR image, using normalized gray-scale correlation. This procedure is the same as that described in the OLI Focal Plane Alignment Calibration Algorithm (Section 4.2.11). Since the expected offsets are small, the TIRS L1T and OLI SWIR image windows are the same size. The correlation procedure yields measured deviations (or correlation failure flag) in the line and sample directions, estimated to subpixel accuracy. These measured LOS deviations are in units of output space pixels.

The deviations measured in output space are converted to differences in LOS along- and across-track angles by mapping the reference point location from output space to TIRS input space and then mapping the search point location from output space to TIRS input space. The mappings are performed using the TIRS LOS projection grid that was used to resample the L1T image, and include the test point elevation interpolated from the input DEM. This three-dimensional output space to input space mapping (3d_ols2ils) is described in the TIRS Resampling Algorithm (Section 4.3.3). This logic is identical to that used for OLI focal plane calibration. Once a TIRS input space location is found for both points, the LOS vectors are calculated for each input sample location using the TIRS LOS model. This is described in the Find LOS section of the TIRS LOS Projection Algorithm (Section 4.3.2).

The angular LOS offsets in TIRS input space are then:

$$\text{along track offset} = \frac{r_x}{r_z} - \frac{s_x}{s_z} + (\text{input ref line} - \text{input search line}) * \text{along track IFOV} \quad (1\text{-}1)$$

$$\text{across track offset} = \frac{r_y}{r_z} - \frac{s_y}{s_z} \quad (1\text{-}2)$$

where:

r_x, r_y, r_z = reference x, y, z vector components of LOS
s_x, s_y, s_z = search x, y, z vector components of LOS
input reference line = input line location for reference point
input search line = input line location for search point

4.3.5.6.2 Stage 2: Update – Compute TIRS Alignment Calibration Update

A constrained least squares solution is used to generate the fit between the angular offsets and the corrections to the TIRS alignment angles and per-SCA Legendre polynomial coefficients. Constraints are necessary to separate the alignment angle estimates from the Legendre polynomial coefficients since the angular effects could be largely absorbed by the Legendre polynomials. The constraints are implemented as supplemental observations that enforce relationships between solution parameters.

There are two options for constraining the parameters. The first, default, option is to constrain the Legendre coefficients so that they do not attempt to model the rotation effects. The second option is to constrain the angular corrections to be zero. This effectively reduces the TIRS alignment calibration solution to be a simple focal plane

calibration solution, similar to the corresponding OLI algorithm. Both options use three additional constraint observations (described in more detail below) with an associated 3-by-3 constraint weight matrix. The weight matrix is a diagonal matrix with the diagonal terms containing a common weight, read from the CPF, for all 3 constraints.

Unlike the OLI focal plane calibration procedure, which calibrates one SCA at a time, the TIRS alignment solution is simultaneous so that both the SCA-specific Legendre coefficients and the global alignment angles can be estimated. This also requires that both the along- and across-track Legendre coefficients be solved for at the same time. There are thus 27 unknowns to be solved for: 3 alignment angles + 3 SCAs * (4 along-track Legendre coefficients + 4 across-track Legendre coefficients).

There are N_k tie point observations in SCA k (k=1,2,3), so the total number of observations is $N = N_1 + N_2 + N_3$. The observation matrix is an Nx2 matrix containing the measured X and Y offsets in TIRS input space angular units. Note that, unlike the OLI focal plane calibration, there is a single observation matrix containing both the along-track (X) offsets and the across-track (Y) offsets.

$$\underset{2Nx1}{[B]} = \begin{bmatrix} x\,offset_1 \\ y\,offset_1 \\ x\,offset_2 \\ y\,offset_2 \\ \vdots \\ x\,offset_N \\ y\,offset_N \end{bmatrix} \tag{2-1}$$

The design matrix is a 2Nx27 matrix with each row of the matrix containing the alignment angle partial derivatives and the Legendre polynomial terms associated with the reference sample location of the corresponding tie point measurement. The calculation of these angle partial derivatives and Legendre polynomial terms, as functions of the input sample location, is described below.

$$[A]_{2Nx27} = \begin{bmatrix} A_{0,1} & A_{1,1} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ A_{0,N1} & A_{1,N1} & 0 & 0 \\ A_{0,N1+1} & 0 & A_{1,N1+1} & 0 \\ \vdots & \vdots & \vdots & \vdots \\ A_{0,N1+N2} & 0 & A_{1,N1+N2} & 0 \\ A_{0,N1+N2+1} & 0 & 0 & A_{1,N1+N2+1} \\ \vdots & \vdots & \vdots & \vdots \\ A_{0,N1+N2+N3} & 0 & 0 & A_{1,N1+N2+N3} \end{bmatrix} \tag{2-2}$$

Where A_0 and A_1 are submatrices defined as:

$$A_{0,j} = \begin{bmatrix} 0 & -1 & y'_j \\ 1 & 0 & -x'_j \end{bmatrix}$$

$$A_{1,j} = \begin{bmatrix} l_{0,j} & l_{1,j} & l_{2,j} & l_{3,j} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & l_{0,j} & l_{1,j} & l_{2,j} & l_{3,j} \end{bmatrix}$$

And where:
x'_j = the x coordinate of tie point j, evaluated using the current estimate of the Legendre polynomials.
y'_j = the y coordinate of tie point j, evaluated using the current estimate of the Legendre polynomials.
$l_{i,j}$ = i^{th} Legendre polynomial term associated with the tie point location of the j^{th} measurement.

The Legendre polynomial terms (contained in the 2x8 $A_{1,j}$ submatrix) will be in the set of 8 columns associated with the SCA containing tie point j. Thus, columns 4-11 are associated with SCA 1, columns 12-19 are associated with SCA 2, and columns 20-27 are associated with SCA 3. In equation (2-2) tie points 1 through N_1 fall in SCA 1, points N_1+1 through N_1+N_2 fall in SCA 2, and points N_1+N_2+1 through N (=N_1+N_2+N_3) fall in SCA 3. Note that the partial derivative of the X offset with respect to the yaw correction is the tie point Y coordinate and the partial derivative of the Y offset with respect to the yaw correction is the –X tie point coordinate. Thus, the y'_j coordinate appears in the first (X observation) row of the A_0 submatrix and the $-x'_j$ coordinate appears in the second (Y observation) row of the A_0 submatrix.

The tie point observations are weighted by a diagonal weight matrix. The weight matrix [$\mathbf{W_t}$] is a 2Nx2N diagonal matrix where the diagonal elements are the tie point weight value read from the CPF.

$$[W_t]_{2Nx2N} = \begin{bmatrix} w_t & 0 & \cdots & 0 \\ 0 & & \ddots & \vdots \\ \vdots & & & 0 \\ 0 & \cdots & 0 & w_t \end{bmatrix} \qquad (2\text{-}3)$$

where w_t = tie point weight

The weight matrix is included to make it possible to differentially weight the measured deviations based on correlation strength, but this is not implemented in the baseline algorithm. Instead, a common weight, read from the CPF, is used for all points. This weight value provides a relative weighting of the tie point observations relative to the constraints.

The form of the constraint observations depends upon the constraint option selected. For the default case, the Legendre coefficient adjustments are constrained to prevent them from modeling angular alignment effects. This is done by considering the effects of angular alignment changes at the center of each SCA. The x and y coordinates at the center of SCA k are as follows:

$$x_0'_k = ax_{0k} - ax_{2k} / 2 \qquad (2\text{-}4)$$
$$y_0'_k = ay_{0k} - ay_{2k} / 2$$

where:

ax_{0k} and ax_{2k} are the first and third x Legendre coefficients for SCA k, and
ay_{0k} and ay_{2k} are the first and third y Legendre coefficients for SCA k.

Therefore, the Legendre corrections at the center of each SCA are as follows:

$$\Delta x_0'_k = \Delta ax_{0k} - \Delta ax_{2k} / 2 \qquad (2\text{-}5)$$
$$\Delta y_0'_k = \Delta ay_{0k} - \Delta ay_{2k} / 2$$

where:

Δax_{0k} and Δax_{2k} are the first and third x Legendre corrections for SCA k,
Δay_{0k} and Δay_{2k} are the first and third y Legendre corrections for SCA k.

To control the modeling of alignment effects in the Legendre polynomials we constrain the Legendre adjustments to the locations of the SCA center points such that the mean center point x (pitch) and y (roll) locations do not change and such that the x (along-track) coordinates of the center points of the two outboard SCAs (1 and 3) do not change in opposite directions (yaw):

$$\begin{aligned} &\Delta ay_{01} + \Delta ay_{02} + \Delta ay_{03} - (\Delta ay_{21} + \Delta ay_{22} + \Delta ay_{23})/2 = 0 && \text{(roll)} \\ &\Delta ax_{01} + \Delta ax_{02} + \Delta ax_{03} - (\Delta ax_{21} + \Delta ax_{22} + \Delta ax_{23})/2 = 0 && \text{(pitch)} \\ &\Delta ax_{01} - \Delta ax_{21}/2 - \Delta ax_{03} + \Delta ax_{23} / 2 = 0 && \text{(yaw)} \end{aligned} \qquad (2\text{-}6)$$

The corresponding constraint matrix is:

$$[C]_{3x27} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & 1 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad (2\text{-}7)$$

The second constraint option fixes the alignment angle adjustments at zero, allowing all adjustments to be modeled through the Legendre coefficients. The corresponding constraint matrix is:

$$[C]_{3x27} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad (2\text{-}8)$$

The constraint weight matrix [W_c] is a 3-by-3 diagonal matrix containing the constraint weight read from the CPF:

$$[W_c]_{3x3} = \begin{bmatrix} w_c & 0 & 0 \\ 0 & w_c & 0 \\ 0 & 0 & w_c \end{bmatrix} \quad (2\text{-}9)$$

where w_c = constraint weight

Note that the observation matrix is zero for the constraints no matter which constraint option is selected.

The solution for the updates to the TIRS alignment angles and to the Legendre LOS coefficients can be found from:

$$[\boldsymbol{\theta}] = ([A]^T[W_t][A] + [C]^T[W_c][C])^{-1}([A]^T[W_t][B]) \quad (2\text{-}10)$$

The matrix [θ] is a 27x1 vector containing the corrections to be applied to the alignment angles and to the Legendre coefficients. These corrections are added to the original alignment angles and Legendre LOS coefficients to compute the updated TIRS alignment parameters.

$$[\boldsymbol{\theta}] = \begin{bmatrix} \Delta r \\ \Delta p \\ \Delta y \\ [\Delta ax_1] \\ [\Delta ay_1] \\ [\Delta ax_2] \\ [\Delta ay_2] \\ [\Delta ax_3] \\ [\Delta ay_3] \end{bmatrix} \quad [\Delta ax_k] = \begin{bmatrix} \Delta ax_{0k} \\ \Delta ax_{1k} \\ \Delta ax_{2k} \\ \Delta ax_{3k} \end{bmatrix} \qquad [\Delta ay_k] = \begin{bmatrix} \Delta ay_{0k} \\ \Delta ay_{1k} \\ \Delta ay_{2k} \\ \Delta ay_{3k} \end{bmatrix} \quad (2\text{-}11)$$

The 10.8 micrometer band is used for TIRS alignment. TIRS band calibration uses the 10.8 micrometer band as the reference for the 12.0 micrometer band. A TIRS band alignment calibration should be performed following an update to the TIRS alignment calibration to avoid degrading the band-to-band registration.

Figure 4-74 shows the architecture for the setup portion of the TIRS Alignment Calibration algorithm.

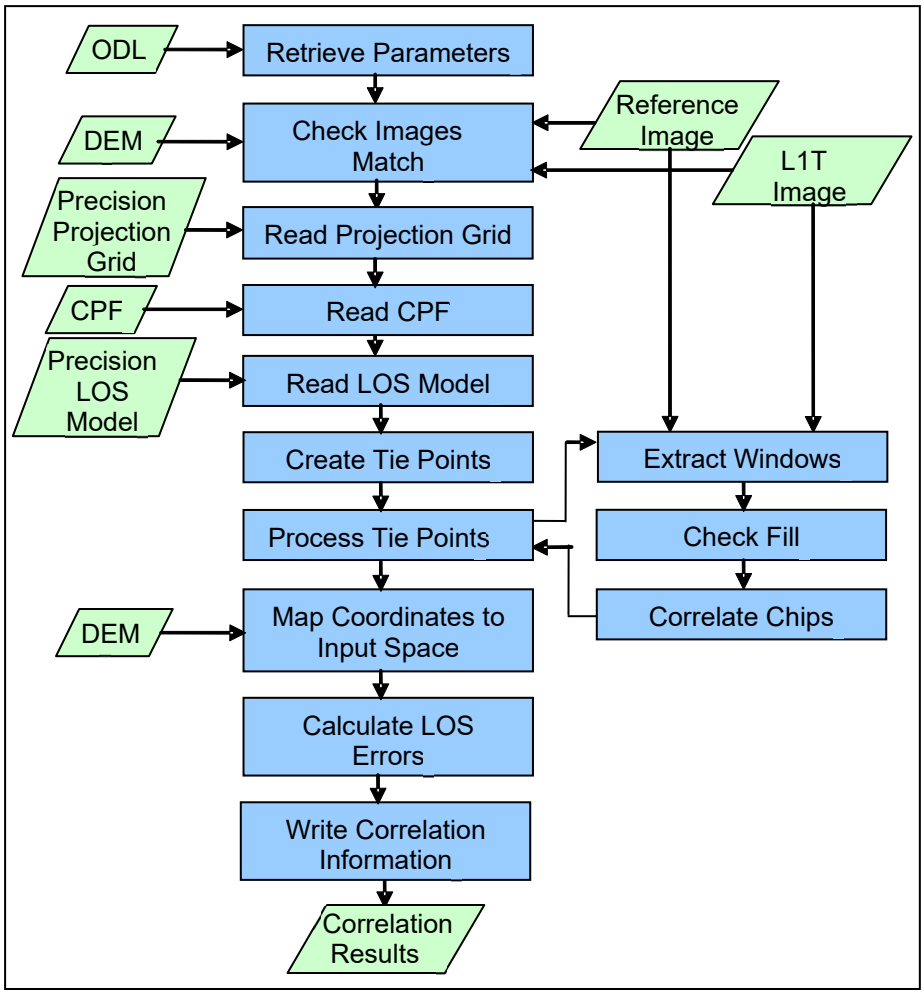

Figure 4-74. TIRS Alignment Calibration Setup Algorithm Architecture

Figure 4-75 shows the architecture of the alignment solution portion of the TIRS Alignment Calibration algorithm.

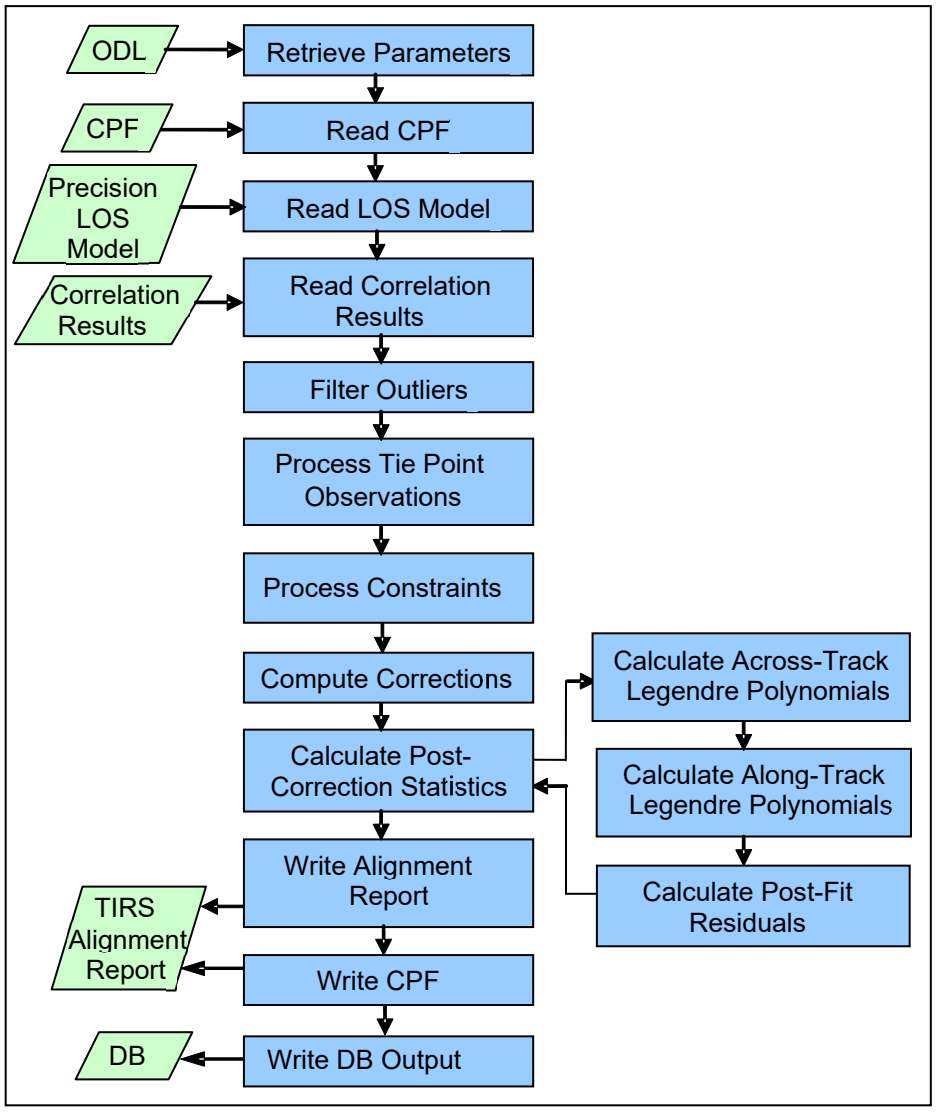

Figure 4-75. TIRS Alignment Calibration Update Algorithm Architecture

TIRS Alignment Cal Setup Sub-Algorithm (TIRS_Align_Setup)
This routine is the main driver for the setup portion of the TIRS alignment calibration. The setup portion consists of correlating points between the search TIRS image and the reference OLI SWIR image, and converting the correlation offsets into line-of-sight deviations that can be used to correct each SCA's detector array, modeled by a cubic Legendre polynomial. The results of this program are used in the second portion of TIRS alignment calibration, the generation of new TIRS-to-OLI alignment angles and TIRS SCA Legendre polynomials. This program creates a temporary output file that is read by the second portion of TIRS alignment calibration.

Get TIRS Alignment Parameters Sub-Algorithm (get_tirs_align_parms)
This function gets the input parameters from the input parameter files.

Check Images Match Sub-Algorithm (check_images_match)
This function checks to make sure the L1T TIRS search, OLI L1T SWIR reference, and DEM images all match. The corners of all the images should match to within half a pixel, and the reference and L1T search image should be the same resolution (pixel size) and the same size. This function is an initial check to make sure that all the images are consistent before correlation is attempted. This function assumes the OLI SWIR reference image and the DEM are one-band images.

Set Up Grid Sub-Algorithm (set_up_grid)
This function reads the TIRS grid file into the grid data structure. The whole grid structure is returned so the caller can free all memory allocated when the grid was read using the grid deallocation call.

Select Correlation Points Sub-Algorithm (select_corr_pts)
This function selects nominal correlation points evenly distributed about the center point of each grid cell (output space). To ensure evenly distributed tie point locations during correlation, locations are defined to lie at the center of each resampling grid cell, or sub-cell. There will be pts_per_cell equally-spaced points per grid cell. For example, if there are 4 points per cell, they will be placed as shown in Figure 4-76.

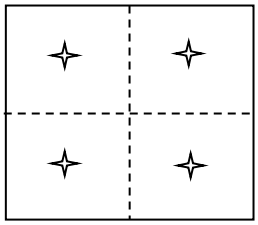

Figure 4-76. Correlation Point Placement in Grid Cell

The cell is divided into a 2-by-2 grid of 4 sub-cells, and each sub-cell is divided in half to place the point in the middle, yielding points at (0.25,0.25), (0.75,0.25), (0.25,0.75), and (0.75,0.75).

The calculation of the output space line/sample coordinates of the tie points is done as follows:

a) Compute number of rows and columns of tie points in each cell.
   ncol = (int)ceiling(sqrt(pts_per_cell))
   nrow = (int)ceiling((double)pts_per_cell/(double)ncol)
   This creates an array of tie points containing at least pts_per_cell points.

b) For each tie point, i = 1 to ncol and j = 1 to nrow:
   b1) Compute the grid cell fractional location (cfrac,rfrac).

$$cfrac = \frac{2i-1}{2\ ncol} \qquad rfrac = \frac{2j-1}{2\ nrow}$$

   b2) Compute the output space line, ol_{ij}, using bilinear interpolation on the output line numbers at the grid cell corners, where l_{UL}, l_{UR}, l_{LL}, and l_{LR} are the output space line coordinates at the grid cell upper-left, upper-right, lower-left, and lower-right corners, respectively:

$ol_{ij} = l_{UL} * (1-cfrac)*(1-rfrac)$
$+ l_{UR} * cfrac * (1-rfrac)$
$+ l_{LL} * (1-cfrac) * rfrac$
$+ l_{LR} * cfrac * rfrac$

b3) Compute the output space sample, os_{ij}, using bilinear interpolation on the output sample numbers at the grid cell corners, where s_{UL}, s_{UR}, s_{LL}, and s_{LR} are the output space sample coordinates at the grid cell upper-left, upper-right, lower-left, and lower-right corners, respectively:

$os_{ij} = s_{UL} * (1-cfrac)*(1-rfrac)$
$+ s_{UR} * cfrac * (1-rfrac)$
$+ s_{LL} * (1-cfrac) * rfrac$
$+ s_{LR} * cfrac * rfrac$

Note that the bilinear weights are the same for the line and sample computations and only need be computed once.

Calculate Input Space Errors Sub-Algorithm (calc_input_space_errors)
This function calculates the errors in TIRS input space pixels. This is done by first correlating in output space and converting the correlated locations to input space.

Perform Correlation Sub-Algorithm (perform_correlation)
This function performs the normalized gray-scale correlation at each point. It does this by invoking the correlation library routines described in the GCP Correlation Algorithm (Section 4.1.6).

Map Coordinates to Input Space Sub-Algorithm (map_coords_to_input_space)
This function uses the inverse mapping coefficients in the grid to calculate the TIRS input space line/sample for each output space line/sample. It does this for both the reference (OLI SWIR) image line/sample location and the search (TIRS L1T) image line/sample location, mapping both to TIRS input space.

a) For each SCA

a1) For each tie point

a1.1) Interpolate a height from the DEM at the location corresponding to the tie point reference image line/sample coordinates (xxx_get_elevation).

a1.2) Map the reference output line/sample location to its corresponding input line/sample location using axx_3d_ols2ils routine

a1.3) Interpolate a height from the DEM at the location corresponding to the tie point search image line/sample coordinates (xxx_get_elevation).

a1.4) Map the search output line/sample location to its corresponding input line/sample location using the axx_3d_ols2ils

Calculate LOS Errors Sub-Algorithm (calc_los_errors)
This function uses the tie-point reference and searches TIRS input space locations to calculate the angular line-of-sight errors.

b) For each SCA

b1) For each tie point

b1.1) Calculate reference line of sight vector for sample location using the nominal detector type and the precision LOS model (axx_findlos).

b1.2) Calculate the search LOS vector for sample location using the nominal detector type and the precision LOS model.

b1.3) Calculate the deviations in terms of the difference in the LOS along and across-track angles:

along-track LOS error = ref los.x/los.z – srch los.x/los.z
+ input line error * along-track IFOV

across-track LOS error = ref los.y/los.z – srch los.y/los.z

Output Correlation Information Sub-Algorithm (output_correlation_info)
This function writes the tie point correlation results to a file. The correlation points are dumped to a binary file, so the second phase of TIRS alignment calibration (TIRS alignment update) can read them directly back in. First, a long integer is written to indicate the number of records, and then all the records are written. Each record contains the fields seen in Table 4-42.

Type	Field	Description
int	sca_number	SCA number (0-relative)
int	grid_column	grid column number (0-relative)
int	grid_row	grid row number (0-relative)
double	nom_os_pt.line	nominal output space point line
double	nom_os_pt.samp	nominal output space point sample
double	ref_os_pt.line	reference output space line
double	ref_os_pt.samp	reference output space sample
double	srch_os_pt.line	search output space line
double	srch_os_pt.samp	search output space sample
double	ref_is_pt.line	reference TIRS input space line
double	ref_is_pt.samp	reference TIRS input space sample
double	srch_is_pt.line	search TIRS input space line
double	srch_is_pt.samp	search TIRS input space sample
double	los_err.line	angular along-track LOS error
double	los_err.samp	angular across-track LOS error
double	los_err_pix.line	line LOS error in pixels
double	los_err_pix.samp	sample LOS error in pixels
double	correlation_accuracy	correlation accuracy
ActiveFlag	active_flag	correlation success flag
double	pt_weight	point weight for use in fit

Type	Field	Description
double	fit_residual.line	line residual from fit of pts
double	fit_residual.samp	sample residual from fit of pts

Table 4-42. Tie Point Correlation Results File Format

This is not a human-readable (ASCII) file, because it is only used to transport information from the first phase of calibration to the second. If the file already exists, it will be overwritten.

TIRS Alignment Update Sub-Algorithm (Generate_Legendre_Polynomials)
This routine is the main driver for the least squares solution and alignment update portion of TIRS alignment calibration. The TIRS alignment update portion reads the results of the TIRS alignment setup, filters the outliers, fits the data to a set of angular alignment corrections and Legendre polynomial corrections, updates the TIRS-to-OLI alignment and TIRS SCA models, and generates output reports. This process is outlined below.

a) Initialize the normal equation matrix [N] (27x27) and constant vector [L] (27x1) to zero.

b) Process each tie point (j)

b.1) Build design matrix [A_j].

Calculate normalized detector for reference sample location. Note that in this context the “detector” number is the input sample number within the SCA containing the tie point (SCA #k).

$$\text{normalized detector} = \frac{2 * \text{detector}}{(\text{number of detectors} - 1)} - 1$$

where:
detector = reference sample location (0 ... Ndet-1)
number of detectors = number of detectors in current SCA

Calculate two rows of design matrix associated with current tie point:

$l_{0,j}$ = 1
$l_{1,j}$ = normalized detector
$l_{2,j}$ = (3*(normalized detector)^2– 1) / 2
$l_{3,j}$ = normalized detector *(5 *(normalized detector)^2– 3) / 2
where j = tie point number

$$x'_j = \sum_{i=0}^{3} (ax_{ik}) l_{i,j}$$

$$y'_j = \sum_{i=0}^{3} (ay_{ik}) l_{i,j}$$

$$\underset{2x27}{[A_j]} = [[A_0] \quad [A_1] \quad [A_2] \quad [A_3]]$$

where:

$$\underset{2x3}{[A_0]} = \begin{bmatrix} 0 & -1 & y'_j \\ 1 & 0 & -x'_j \end{bmatrix}$$

$$\underset{2x8}{[A_i]} = \begin{bmatrix} l_{0,j} & l_{1,j} & l_{2,j} & l_{3,j} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & l_{0,j} & l_{1,j} & l_{2,j} & l_{3,j} \end{bmatrix} \quad \text{if k = i}$$

$$\underset{2x8}{[A_i]} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{otherwise}$$

b.2) Build weight matrix [W_j] using the (fixed) input weight value.

$$\underset{2x2}{[W_j]} = \begin{bmatrix} w_t & 0 \\ 0 & w_t \end{bmatrix}$$

b.3) Build the observation matrix [B_j] from the measured x and y offsets in terms of angular differences.

$$\underset{2x1}{[B_j]} = \begin{bmatrix} x\,offset_j \\ y\,offset_j \end{bmatrix}$$

b.4) Add this observation to the normal equations matrix [N] and to the constant vector [L].

$[N] = [N] + [A_j]^T [W_j] [A_j]$
$[L] = [L] + [A_j]^T [W_j] [B_j]$

c) Add the constraints.

c.1) Form the constraint design matrix [C] based upon the user-selected constraint option, using equation (2-7) or equation (2-8) above.

c.2) Form the constraint weight matrix [W_c] per equation (2-9) above.

c.3) Add the constraint contribution to the normal equations matrix [N]. Note that there is no constraint contribution to the constant vector [L].

$[N] = [N] + [C]^T [W_c] [C]$

d) Solve for alignment angle and Legendre coefficient corrections using weighted least-squares routine (see the Fit Parameters sub-algorithm below).

e) Calculate pre-fit statistics from the original measured deviations.

e.1) Calculate statistics for along- and across-track offsets used in b.3.

Compute mean, standard deviation and RMSE for the offsets, grouping by SCA. Values are calculated for along- and across-track directions independently. For each SCA k=1,2,3:

For along-track offsets in SCA k:
e.1.1) Calculate x mean
e.1.2) Calculate x standard deviation
e.1.3) Calculate x RMSE

For across-track offsets in SCA k:
e.1.4) Calculate y mean
e.1.5) Calculate y standard deviation
e.1.6) Calculate y RMSE

f) Calculate post fit residual statistics for correction coefficients

Post-fit residuals are calculated by updating the original measured offsets/deviations used in step b.3 above using the alignment angle and Legendre polynomial corrections. The differences between the original measurements and the offsets modeled by the correction parameters are the residuals. The post-fit statistics are calculated on these residuals.

f.1) For each tie point

f.1.1) Calculate normalized detector for reference sample location (as shown in b.1) and construct the design matrix as shown in b.1.

f.1.2) Calculate the modeled LOS angle corrections by multiplying the design matrix by the [θ] solution vector.

f.1.3) Find the residuals as the difference between the original angular offsets and the LOS angle corrections from f.1.2.

f.2) Calculate statistics for the along- and across-track residuals calculated in f.1.

Compute mean, standard deviation and RMSE for residuals, grouping by SCA. Values are calculated for along- and across-track directions independently. For each SCA k=1,2,3:

For along-track residuals:
f.2.1) Calculate x mean
f.2.2) Calculate x standard deviation
f.2.3) Calculate x RMSE

For across-track residuals:
f.2.4) Calculate y mean
f.2.5) Calculate y standard deviation
f.2.6) Calculate y RMSE

g) For each SCA, add the correction coefficients to original Legendre LOS coefficients (see note #4):

new along $\text{legendre}_{i,sca}$ = update along $\text{legendre}_{i,sca}$ + old along $\text{legendre}_{i,sca}$

new across $\text{legendre}_{i,sca}$ = update across $\text{legendre}_{i,sca}$ + old across $\text{legendre}_{i,sca}$

where:
i = 0,1,2,3 Legendre polynomial number
sca = SCA number

h) Use the computed roll, pitch, and yaw alignment corrections to update the TIRS-to-OLI rotation matrix.

h.1) Compute the delta rotation matrix [∆M] from the ∆r, ∆p, and ∆y corrections.

$$[\Delta M] = \begin{bmatrix} \cos(\Delta p)\cos(\Delta y) & \sin(\Delta r)\sin(\Delta p)\cos(\Delta y) + \cos(\Delta r)\sin(\Delta y) & \sin(\Delta r)\sin(\Delta y) - \cos(\Delta r)\sin(\Delta p)\cos(\Delta y) \\ -\cos(\Delta p)\sin(\Delta y) & \cos(\Delta r)\cos(\Delta y) - \sin(\Delta r)\sin(\Delta p)\sin(\Delta y) & \cos(\Delta r)\sin(\Delta p)\sin(\Delta y) + \sin(\Delta r)\cos(\Delta y) \\ \sin(\Delta p) & -\sin(\Delta r)\cos(\Delta p) & \cos(\Delta r)\cos(\Delta p) \end{bmatrix}$$

h.2) Combine the delta rotation matrix [∆M] with the original rotation matrix [TIRS2OLI], derived from the ACS-to-OLI and ACS-to-TIRS matrices in the CPF, to form the updated rotation matrix [TIRS2OLI]'.

$[TIRS2OLI] = [ACS2OLI]\,[ACS2TIRS]^T$

$[TIRS2OLI]' = [TIRS2OLI]\,[\Delta M]$

h.3) Compute original and updated TIRS-to-OLI alignment angles.

$Roll = \text{atan}(-[TIRS2OLI]_{3,2} / [TIRS2OLI]_{3,3})$
$Pitch = \text{asin}([TIRS2OLI]_{3,1})$
$Yaw = \text{atan}(-[TIRS2OLI]_{2,1} / [TIRS2OLI]_{1,1})$

$Roll' = \text{atan}(-[TIRS2OLI]'_{3,2} / [TIRS2OLI]'_{3,3})$
$Pitch' = \text{asin}([TIRS2OLI]'_{3,1})$
$Yaw' = \text{atan}(-[TIRS2OLI]'_{2,1} / [TIRS2OLI]'_{1,1})$

h.4) Compute the updated ACS-to-TIRS rotation matrix.

Compute the updated OLI-to-TIRS rotation matrix as the transpose of the updated TIRS-to-OLI matrix:

$$[OLI2TIRS]' = [TIRS2OLI]'^{T}$$

Compute the updated ACS-to-TIRS rotation matrix using the ACS-to-OLI matrix from the CPF and the updated OLI-to-TIRS rotation matrix:

$$[ACS2TIRS]' = [OLI2TIRS]' [ACS2OLI]$$

Read Correlation Information Sub-Algorithm (read_correlation_info)
This function reads the correlation information from the file generated by the Output Correlation Information sub-algorithm above.

Filter Outliers Sub-Algorithm (filter_outliers)
This function separates the focal plane correlation data into groups for each SCA for the X (sample) and Y (line) directions. It then finds the standard deviation for the points in each group. Outlier rejection is then performed on the points based on the tolerance selected by the user and the Student's T distribution. This procedure is described in the Geometric Accuracy Assessment Algorithm (Section 4.2.6).

Calculate Point Weights Sub-Algorithm (calculate_point_weights)
This function calculates the weight associated with each correlation point for doing the Legendre polynomial fit.

Currently, this routine assigns the weight passed in to each point, effectively assigning each point an equal weight. Originally, it was thought that the correlation strength would factor into the weight, but that was determined to not be needed. This routine was left in to allow point-specific weight factors to be added at a later date.

Fit Parameters Sub-Algorithm (fit_polynomials)
This function performs the weighted least-squares fit of the correlation data points (using the angular error) to find the alignment angle corrections and the Legendre error polynomials. The least squares correction parameter vector $[\theta]$ is given by solving:

$$[N] [\theta] = [L]$$

Where:
[N] is the [27 x 27] normal equations matrix

$[\theta]$ is the [27 x 1] unknown vector containing the alignment angle and Legendre coefficient corrections we are looking for

[L] is the [27 x 1] constant vector

Solving the above equation for $[\theta]$ yields: $[\theta] = ([N])^{-1} * [L]$

Calculate Post-fit Residuals Sub-Algorithm (calculate_post_fit_residuals)
This function calculates the residual statistics, as described in step f) above, after the alignment corrections and Legendre polynomial coefficient corrections have been calculated.

Calculate Legendre Polynomial Sub-Algorithm (calc_legendre_poly)
This function calculates the Legendre polynomial for the input normalized detector value, x:

along = $coeff\_along_0 + coeff\_along_1\ x + coeff\_along_2\ (3*x^2 - 1)/2 + coeff\_along_3\ x*(5*\ x^2 - 3)/2$

across = $coeff\_across_0 + coeff\_across_1\ x + coeff\_across_2\ (3*x^2 - 1)/2 + coeff\_across_3\ x*(5*\ x^2 - 3)/2$

Create TIRS Alignment Report Sub-Algorithm (create_tirs_alignment_report)
This function generates a file reporting the results of the TIRS-to-OLI alignment angle and TIRS SCA Legendre polynomial fit calculations. The report file contents are shown in Table 4-43 below.

Write Coefficients Sub-Algorithm (write_coeffs)
This function writes an entire set of coefficients to the indicated output file.

Write TIRS Alignment Calibration Results to Characterization Database (trend_to_database)
This function writes the results of the TIRS-to-OLI alignment angle and TIRS SCA Legendre polynomial fit calculations to the geometric characterization database. The output is only written to the database if the post-fit along- and across-track RMSE statistics are all below the threshold values specified in the CPF (the trending metrics). The characterization database output is listed in Table 4-43 below.

Write SCA Parameters CPF Sub-Algorithm (write_SCA_parameters_cpf)
This function writes the updated Legendre coefficients to a new LOS_LEGENDRE CPF parameter group, in the ODL format used by the CPF, to a separate ASCII output file. All the CPF LOS_LEGENDRE parameter values except for the new Legendre coefficients are extracted from the original CPF or the LOS Model structure. Current plans call for actual calibration updates to be based on multiple scene results extracted from the characterization database, so this capability is primarily a convenience for testing purposes.

Write TIRS-to-OLI Alignment Parameters CPF Sub-Algorithm (update_alignment_parameters_cpf)
This function writes the ATTITUDE_PARAMETERS parameter group of the CPF, in the ODL format used by the CPF, to a separate ASCII output file. The updated ACS-to-TIRS rotation matrix computed in h.4) above is written to the parameter group along with the current values for all other parameters in that group. Current plans call for actual calibration updates to be based on multiple scene results extracted

from the characterization database, so this capability is primarily a convenience for testing purposes.

Algorithm Output Details

The contents of the output TIRS alignment calibration report file and the corresponding geometric characterization database outputs are summarized in Table 4-43 below. All fields are written to the output report file but only those with "Yes" in the "Database Output" column are written to the characterization database. Note that the first eleven fields listed constitute the standard report header.

Field	Description	Database Output
Date and time	Date (day of week, month, day of month, year) and time of file creation.	Yes
Spacecraft and instrument source	Landsat 8/9 and TIRS	Yes
Processing Center	EROS Data Center SVT	Yes
Work order ID	Work order ID associated with processing (blank if not applicable)	Yes
WRS path	WRS path number	Yes
WRS row	WRS row number	Yes
Software version	Software version used to create report	Yes
Off-nadir angle	Scene off-nadir roll angle (in degrees) (only nadir-viewing scenes are used for TIRS alignment)	Yes
Acquisition type	Earth, Lunar, or Stellar (only Earth-viewing scenes are used for TIRS alignment calibration)	Yes
Geo Char ID	Geometric Characterization ID	Yes
L1T image file	Name of TIRS L1T used to measure tie points	No
Acquisition date	Date of L1T image acquisition (new)	Yes
Reference image file	Name of reference (OLI) image used to measure tie points	Yes
Original TIRS-to-OLI angles	Original TIRS-to-OLI roll-pitch-yaw alignment angles in radians (new)	Yes
TIRS-to-OLI correction angles	Estimated roll-pitch-yaw corrections to the TIRS-to-OLI alignment knowledge in radians (new)	Yes
Update TIRS-to-OLI angles	Updated TIRS-to-OLI roll-pitch-yaw alignment angles in radians (new)	Yes
Original TIRS alignment matrix	Original 3x3 TIRS-to-OLI alignment matrix (new)	No
Updated TIRS alignment matrix	Updated 3x3 TIRS-to-OLI alignment matrix (new)	No
Confidence Level	Confidence level used for outlier rejection	Yes
Fit Order	Order of Legendre fit	Yes
Number of SCAs	Number of SCAs calibrated (3)	Yes
For each SCA:		
SCA Number	Number of the current SCA (1-3)	Yes
Original AT Legendre coeffs	Original along-track Legendre coefficients: a0, a1, a2, a3	Yes
Original XT Legendre coeffs	Original across-track Legendre coefficients: b0, b1, b2, b3	Yes
Error AT Legendre	The computed updates to the along-track Legendre	Yes

Field	Description	Database Output
coeffs.	coefficients: c0, c1, c2, c3	
Error XT Legendre coeffs.	The computed updates to the across-track Legendre coefficients: d0, d1, d2, d3	Yes
New AT Legendre coeffs	New along-track Legendre coefficients: a'0, a'1, a'2, a'3	Yes
New XT Legendre coeffs	New across-track Legendre coefficients: b'0, b'1, b'2, b'3	Yes
Pre-fit AT statistics	Pre-fit along-track offset mean, standard deviation, and RMSE statistics	Yes
Pre-fit XT statistics	Pre-fit across-track offset mean, standard deviation, and RMSE statistics	Yes
Post-fit AT residual statistics	Post-fit along-track residual mean, standard deviation, and RMSE statistics	Yes
Post-fit XT residual statistics	Post-fit across-track residual mean, standard deviation, and RMSE statistics	Yes
Number of Points	Number of tie points used for current SCA	Yes
CPF Group:	Written to a separate output ODL file	
Effective Date Begin	Beginning effective date of CPF group (from the original CPF): YYYY-MM-DD	No
Effective Data End	Ending effective date of CPF group (from the original CPF): YYYY-MM-DD	No
ACS-to-TIRS rotation matrix	Updated 3x3 attitude control system-to-TIRS rotation matrix	No
Number of SCAs	Number of SCAs (3): Num_SCA = 3	No
For each SCA:		
New Legendre polynomial coefficients	Four (one per band/row) arrays of four along-track Legendre coefficients followed by four arrays of four across-track Legendre coefficients.	No
Tie Point Data:	For each tie point:	
SCA Number	SCA where the tie point was measured	No
Grid Cell Column Number	Column number of the grid cell containing the tie point	No
Nominal Output Space Line	Predicted tie point output space line location	No
Nominal Output Space Sample	Predicted tie point output space sample location	No
LOS Line Error	Measured LOS error delta line (in pixels)	No
LOS Sample Error	Measured LOS error delta sample (in pixels)	No
LOS AT Error	Measured LOS error along-track delta angle (in microradians)	No
LOS XT Error	Measured LOS error across-track delta angle (in microradians)	No
State Flag	Tie point state (outlier) flag	No
AT Fit Residual	Along-track fit residual (in microradians)	No
XT Fit Residual	Across-track fit residual (in microradians)	No

Table 4-43. TIRS Alignment Calibration Output Details

Accessing the TIRS Alignment Results in the Characterization Database
Though not part of the formal TIRS alignment calibration algorithm, some comments regarding the anticipated methods of accessing and analyzing the individual scene TIRS alignment calibration results stored in the characterization database may assist with the design of the characterization database.

The database output from the TIRS alignment calibration algorithm will be accessed by a data extraction tool that queries the characterization database to retrieve TIRS alignment calibration results from multiple scenes. The only processing required on the returned results is to compute the average "new" TIRS-to-OLI alignment angles and the average "new" Legendre coefficients for each SCA across all returned scenes. The returned scene results and computed mean alignment angles and mean Legendre coefficient values will be output in a report containing a comma-delimited table of the retrieved trending results as well as the summary averages.

The geometric results would typically be queried by acquisition date and/or WRS path/row. The most common query would be based on acquisition date range, for example, selecting all of the results for a given calendar quarter:

Acquisition_Date is between 01APR2012 and 30JUN2012

The average alignment angles would be calculated from the updated alignment angles for the individual scenes returned, as:

$$Angle_{j,net} = \frac{1}{numScene} \sum_{i=1}^{numScene} Angle_{j,i}$$

for angle j = roll, pitch, yaw.

The average coefficients would be calculated from the "new" Legendre coefficients for the individual scenes returned, as:

$$Coeff_{SCA,j,net} = \frac{1}{numScene} \sum_{i=1}^{numScene} Coeff_{SCA,j,i}$$

for coefficient j (j=0,1,2,3) for each SCA (SCA=1,2,3).

The query results would be formatted in a set of comma-delimited records (for ease of ingest into Microsoft Excel), one record per scene. Each record would contain all of the "header" fields written to the characterization database (items with "Yes" in the rightmost column of Table 4-43 above) but only the "new" alignment angles and the "new" Legendre coefficients for each SCA. The other fields would be retrieved using general purpose database access tools, if and when desired. A header row containing the field names should precede the database records.

Following the scene records the average alignment angles and Legendre coefficients should be written out in the same CPF/ODL syntax used in the report file. This output

uses the same structure shown in the final row in Table 4-43 above, but contains the average, rather than a single scene's, angles and Legendre coefficients.

4.3.5.7 Notes

Some additional background assumptions and notes include the following:

1. The 10.8 micrometer thermal band will be used to provide the geometric reference for the TIRS instrument. If subsequent band correlation studies show that the 12.0 micrometer band provides superior correlation performance relative to the OLI SWIR bands, it could be used as the reference instead and this decision will be revisited. Such a change would affect the TIRS band alignment calibration algorithm as well.
2. The input TIRS and OLI L1T images are treated as separate inputs in the baseline TIRS algorithm. These could ultimately be contained in the same output image, but since the TIRS image is SCA-separated and the OLI image is SCA-combined it may be best to keep them as two distinct input images even if merged OLI and TIRS images can be produced.
3. The TIRS focal plane model uses third order Legendre coefficients to model the line-of-sight directions for each SCA.
4. The baseline assumption is that Legendre coefficient sets will be stored in the CPF for both active detector rows for each band (10.8 and 12.0 micrometer) on each SCA, but that only the primary detector set will be calibrated. Only the coefficients for the primary row for the 10.8 micrometer band will be updated by this calibration procedure. This assumption may be modified if it is decided that the TIRS CPF will contain only Legendre coefficients for the primary detector rows, in which case there will only be two sets of Legendre coefficients in the CPF, or if the Legendre coefficients for the redundant rows are to be maintained by the calibration procedures, in which case the computed updates will be added to the Legendre coefficients from both rows for the 10.8 micrometer band.
5. The ATTITUDE_PARAMETERS CPF parameter group contains the ACS-to-TIRS and ACS-to-OLI alignment matrices. These two matrices are related by the TIRS-to-OLI alignment matrix, which is maintained by the TIRS alignment calibration algorithm, as follows: [ACS2OLI] = [TIRS2OLI] [ACS2TIRS]. To avoid the redundancy inherent in retaining all three of these matrices in the CPF, the [TIRS2OLI] matrix is constructed, when needed, from the ACS-to-sensor matrices as:
   $$[TIRS2OLI] = [ACS2OLI]\,[ACS2TIRS]^{-1}$$
   This [TIRS2OLI] matrix is updated by the TIRS alignment calibration procedure and the result is then used to update the [ACS2TIRS] matrix in the CPF as:
   $$[ACS2TIRS] = [TIRS2OLI]^{-1}\,[ACS2OLI].$$
6. The LOS_LEGENDRE CPF group contains the TIRS focal plane model in the form of the along-track and across-track Legendre coefficients updated by this algorithm.

4.3.6 TIRS Band Registration Accuracy Assessment

4.3.6.1 Background/Introduction

The TIRS Band Registration Accuracy Assessment (BRAA) Algorithm, or the Band-to-Band (B2B) Characterization process, measures the relative band alignment between the spectral bands on each Sensor Chip Assembly (SCA) for the TIRS instrument. The displacement for every pair-wise combination of all bands requested for assessment is measured on each SCA, creating a set of band-to-band measurements for each SCA. The number of bands available for assessment could be from two (the primary 10.8 and 12 micrometer thermal bands) to four for TIRS only or as many as thirteen - 9 OLI reflective bands and two separate representations of the two TIRS spectral bands derived from the primary and redundant TIRS detector rows - depending on the contents of the input L1TP image(s). The residuals measured from the B2B characterization process will be used to assess the accuracy of the band-to-band registration of the TIRS instrument, and if need be, used as input to the band calibration algorithm in order to calculate new LOS parameters for the CPF.

The B2B characterization process works by choosing tie point locations within band pairs of each SCA, extracting windows of imagery from each band and performing gray-scale correlation on the image windows. Several criteria are used in determining whether the correlation process was successful. These criteria include measured displacement and strength of the correlation peak. The subpixel location of the measured offset is calculated by fitting a 2^{nd} order polynomial around the discrete correlation surface and solving for the fractional peak location of the fitted polynomial. The total offset measured is then the integer location of the correlation peak plus the subpixel location calculated.

There are several options available for processing data through the Band Registration Accuracy Assessment algorithm. These include choosing evenly spaced points for location of the windows extracted, choosing to use the TIRS LOS projection grid for determining window locations in order to avoid fill within the image files, specifying the bands to process, and specifying the valid pixel range to use during correlation. Note that, unlike OLI, in which individual SCAs can be selected for processing (to support the analysis of lunar data), we always process all three TIRS SCAs.

The TIRS B2B characterization algorithm will typically be exercised in one of two different modes, depending upon the input image provided. When operating on SCA-separated TIRS images, tie points based on the TIRS LOS projection grid would be generated to evaluate the internal registration of the TIRS spectral bands. The resulting tie point measurements would be suitable for subsequent use in TIRS band alignment calibration. When operating on SCA-combined images containing both TIRS and OLI data, a regular array of tie points would be used to measure the registration of all selected TIRS and OLI band pair combinations. These tie point results would be suitable for characterizing TIRS to OLI band registration performance.

The TIRS B2B characterization and OLI B2B characterization algorithms are very similar and could potentially be combined at some point in the future. This revision of the TIRS B2B algorithm addresses only the TIRS-only and OLI/TIRS combined cases of Band Registration Accuracy Assessment.

4.3.6.2 Dependencies

The TIRS BRAA assumes that a cloud free Earth viewing L1TP image has been generated and depending on the tie point selection type chosen, that the LOS Model Correction and the LOS Projection and Gridding algorithms have been executed to create a TIRS LOS projection grid file. The L1TP image may be in either the SCA-separated or SCA-combined format, as noted above, and would use either the SOM or UTM path-oriented projection. In any case, the TIRS spectral bands would be resampled to 30m pixel spacing.

4.3.6.3 Inputs

The BRAA and its component sub-algorithms use the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs
ODL file (implementation)
Calibration Parameter File name
TIRS L1TP image file name
OLI L1TP image file name (optional)
TIRS LOS projection grid (optional)
B2B characterization output file
Output residuals file name
Output statistics file name
TIRS bands to process
OLI bands to process (optional)
Processing Parameters
Outlier (t-distribution) threshold
Tie-point type (1 = regularly spaced, 2 = selected using TIRS grid)
Tie-point spacing in line direction
Tie-point spacing in sample direction
Fill range maximum
Fill range minimum
Fill threshold or percentage
Correlation window size lines
Correlation window size samples
Trending flag
Geometric Characterization ID (for trending)
Work Order ID (for trending)
WRS Path (for trending)
WRS Row (for trending)
Calibration Parameter File
Fill range maximum (default value)
Fill range minimum (default value)

Algorithm Inputs
Fill threshold or percentage (default value)
Correlation window size lines (default value)
Correlation window size samples (default value)
Maximum allowable offset
Strength of correlation peak
Correlation Fit method
Trending metrics – standard deviation thresholds per band (see note #3).

4.3.6.4 Outputs

Pan downsampled image (if OLI bands are included)
B2B residuals file (see Table 4-45 below for details)
B2B output data file (see Table 4-44 below for details)
B2B statistics file (see Table 4-46 below for details)
B2B characterization trending (if trend flag set to yes, see Table 4-46 for details)

4.3.6.5 Options

Grid-based tie-point generation
SCA-separated TIRS-only processing or SCA-combined OLI/TIRS processing

4.3.6.6 Procedure

TIRS Band Registration Accuracy Assessment measures the misalignment between the TIRS spectral bands and, optionally, between the TIRS and OLI spectral bands, after all known geometric effects have been taken into account. In the case where an SCA-separated TIRS-only image is used as input, the results from the band registration assessment can be used by the TIRS band alignment calibration routine (See TIRS Alignment Calibration, Section 4.3.5) to estimate new Legendre LOSs (See TIRS Line-of-Sight Model Creation Algorithm, Section 4.3.1) for both TIRS bands for each SCA. If an SCA-combined TIRS and OLI image is used as input, the results would be used solely for band registration accuracy characterization purposes. Due to the different viewing angles for each band of each SCA within TIRS, and between TIRS and OLI, geometric displacement due to relief must be removed from the imagery for band-to-band characterization of Earth acquisitions, i.e., input imagery for band registration assessment must be precision and terrain corrected (See TIRS Resampling Algorithm, Section 4.3.3).

The steps involved in band registration assessment are depicted in Figure 4-77 and include creating datasets with common pixel resolutions (if the OLI panchromatic band is included); choosing locations (tie-point locations) for measurement; performing mensuration; removing outliers from calculated residuals; and calculating statistics from the remaining residuals. Residuals are measured on each SCA for each band combination requested through the input parameters.

4.3.6.6.1 Stage 1 - Data Input

The data input stage involves loading the information required to perform the band registration assessment. This includes reading the image file, retrieving the output B2B file names: output, residuals and statistic files; retrieving or initializing processing parameters: maximum displacement, fill range, fill threshold, minimum correlation peak, t-distribution threshold, bands to process, correlation window size, trending metrics, tie-point method; and if tie-point method is set to grid-based (for SCA-separated input images) the TIRS LOS grid file name will be read. Once the input file, and if need be the TIRS LOS grid name, has been retrieved the files and the information stored within them can be opened and read.

4.3.6.6.1.1 Optional Stage 2 - Creating a Reduced Resolution PAN band (if OLI and TIRS SCA-combined image is input)

Before displacement between the OLI PAN band and the other multispectral bands can be measured the PAN band must be reduced in resolution to match that of the multispectral bands. An oversampled cubic convolution function is used to reduce the resolution of the PAN band. Cubic convolution interpolation uses a set of piecewise cubic spline interpolating polynomials. The polynomials have the following form:

$$f(x) = \begin{cases} (\alpha+2)|x|^3 - (\alpha+3)|x|^2 + 1 & 0 \le |x| < 1 \\ \alpha|x|^3 - 5\alpha|x|^2 + 8\alpha|x| - 4\alpha & 1 \le |x| < 2 \\ 0 & |x| > 2 \end{cases}$$

Since the cubic convolution function is a separable function, a two dimensional representation of the function is given by multiplying two one-dimension cubic convolution functions, one function representing the x-direction the other function representing the y-direction. For an offset of zero, or x = 0, and α = -1.0 the discrete cubic function has the following values; f(0) = 1 and f(n) = 0 elsewhere. Thus convolving the cubic convolution function of x = 0 with a dataset leaves the dataset unchanged.

$$y[n] = f[n] \otimes x[n]$$
$$\text{for } x = 0$$
$$\text{gives } y[n] = x[n]$$
$$\text{where } \otimes \text{ is the convolution operator}$$

Figure 4-77 shows what the cubic function f(t) (dashed line) and the corresponding discrete weights for an offset, or phase, of zero (crossed-dots).

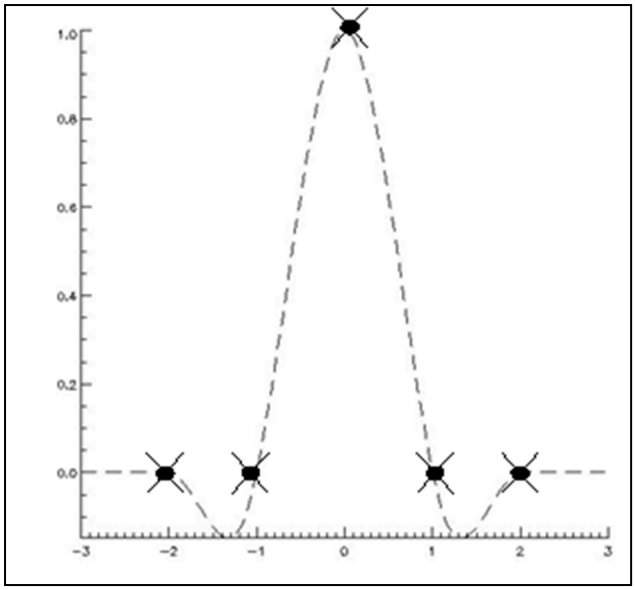

Figure 4-77. Cubic Convolution Function and Weights for Phase of Zero

To spatially scale an input data stream an oversampled cubic convolution function with an offset of x =0 can be used. This can best be understood by looking at the Fourier Transform scaling property of a function that is convolved with a given input dataset:

$$f(t) \otimes x(t) \Leftrightarrow F(\omega) \bullet X(\omega)$$

$$x(at) = \frac{1}{|a|} X\left(\frac{\omega}{a}\right)$$

Where:

$\otimes$ is convolution
● is multiplication
F is the Fourier transform of *f*
X is the Fourier transform of *x*
t is time
ω is frequency

Applying the cubic function and scaling properties to an image data file shows that densifying the points applied with the cubic convolution function will in turn inversely scale the function in the frequency domain, thus reducing the resolution of the imagery. By setting the cubic convolution offset to zero, densifying the number of weights of the cubic function, and convolving these weights to an image file a reduction in resolution will be the resultant output image file. Figure 4-78 shows the cubic function with corresponding weights densified by a factor of two and a phase shift of zero. To ensure that the cubic weights do not scale the DNs of the output imagery during convolution the cubic weights are divided by the scale factor.

$$f_s[n] = \frac{1}{2}\sum_{n=-4}^{4} f\left(\frac{n}{2}\right)$$

Where:

f_s[n] = scaled cubic convolution weights

f(n) = cubic convolution function

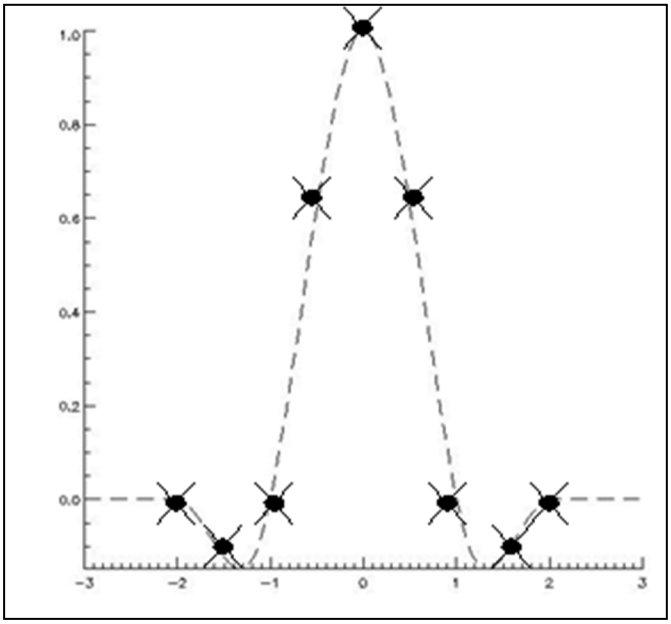

Figure 4-78. Cubic Convolution Densified by a Factor of 2

Scaling the cubic convolution function by a factor of 2 gives the following 1-dimensional set of weights:

$$ccw[n] = \begin{bmatrix} 0.0 & -0.0625 & 0.0 & 0.3125 & 0.5 & 0.3125 & 0.0 & -0.0625 & 0.0 \end{bmatrix}$$

To determine the 2-dimensional cubic convolution weights two 1-dimensional sets of cubic weights are multiplied together (note only 7 values are needed for ccw, outside of this extent the weights are zero):

$$ccw[n] \times ccw[m] = ccw[n,m] = \begin{bmatrix} 0.0039 & 0.0 & -0.0195 & -0.0313 & -0.0195 & 0.0 & 0.0039 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.0195 & 0.0 & 0.0977 & 0.1563 & 0.0977 & 0.0 & -0.0195 \\ -0.0313 & 0.0 & 0.1563 & 0.25 & 0.1563 & 0.0 & -0.0313 \\ -0.0195 & 0.0 & 0.0977 & 0.1563 & 0.0977 & 0.0 & -0.0195 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0039 & 0.0 & -0.0195 & -0.0313 & -0.0195 & 0.0 & 0.0039 \end{bmatrix}$$

Where:

ccw[n] is a 8x1 1-dimensional set of cubic weights
ccw[m] is a 1x8 1-dimensional set of cubic weights

4.3.6.6.1.1.1 Procedure for Reducing PAN band

To reduce the resolution of the OLI PAN band apply the ccw[n,m] weights to the PAN image data:

$$\text{reduced pan} = ccw[n,m] * \text{pan band}$$

Note: number of lines and number of samples listed below pertain to the size of the PAN band imagery.

Reduce PAN Band Resolution Processing Steps

1. Set line =0 then for every other PAN line

    1.1. Set sample = 0 then for every other PAN sample

    1.2. initialize summing variable sum = 0.0

    1.3. For m = -4 to 4

        1.3.1. For n = -4 to 4

        1.3.2. Check to see if current image index is within valid imagery

        1.3.3. if m + line < 0 then line index = -m - line
        else if m + line >= number of lines then line index =
        2 * number of lines - m - line - 1
        else line index = m + line

        1.3.4. if n + sample < 0 then sample index = -n - sample
        else if n + sample >= number of sample then sample index =
        2 * number of samples - n - sample - 1
        else sample index = n + sample

        1.3.5. sum = sum + ccw[n+4,m+4] • pan[line index, sample index]

    1.4. Store output DN for reduced PAN
    output line = line / 2
    output sample = sample / 2
    reduce pan[output line,output sample] = sum

4.3.6.6.1.2 Stage 3 - Create Tie-point Locations

Tie point locations may be determined in an evenly spaced pattern in output space or they may be established in an evenly spaced pattern in input space, using the TIRS LOS projection grid. The first method is used when SCA-combined OLI/TIRS images are input, and the second is used when SCA-separated TIRS-only images are input.

4.3.6.6.1.2.1 Determine Evenly Spaced Tie-points (See notes #6 and #7)

To determine evenly spaced tie-point locations a tie-point location is defined by stepping through the output space of the imagery by the user defined steps N,M.

Create Evenly Spaced Tie-Points Processing Steps

1. Determine number of tie-points in sample and line direction:

$$\text{tie - point spacing x} = \frac{\text{ONS} - \text{correlation window samples}}{\text{M - 1}}$$

$$\text{tie - point spacing y} = \frac{\text{ONL} - \text{correlation window lines}}{\text{N - 1}}$$

Where:

M = user entered number of tie-points in sample direction
N = user entered number of tie-points in line direction
ONS = number of samples in output space of multispectral band
ONL = number of lines in output space of multispectral band
Correlation window samples = user entered correlation window size in samples
Correlation window lines = user entered correlation window size in lines

2. Set evenly spaced tie-point locations
   2.1. For j = 0 to N-2

$$\text{tie-point location y}[j] = \frac{\text{correlation window lines}}{2} + j * \text{tie-point spacing y}$$

   2.2. $\text{tie-point location y}[N-1] = ONL - \frac{\text{correlation window lines}}{2}$

   2.3. For i = 0 to M-2

$$\text{tie -point location x}[i] = \frac{\text{correlation window samples}}{2} + i * \text{tie-point spacing x}$$

   2.4. $\text{tie -point location x[M-1]} = ONS - \frac{\text{correlation window samples}}{2}$

4.3.6.6.1.2.2 Determine TIRS Grid Spaced Tie-points (See notes #6 and #7)

For descriptions of the format and data stored within the TIRS LOS grid see the TIRS LOS Projection/Grid Generation Algorithm (Section 4.3.2).

Input Space Tie-points Processing steps.

1. Read image extent parameters from TIRS LOS grid
   INS = input (raw) space number of samples
   INL = input (raw) space number of lines

2. Determine number of tie-points in sample and line direction:

$$\text{spacing x} = \frac{\text{INS} - \text{correlation window samples}}{\text{M - 1}}$$

$$\text{spacing y} = \frac{\text{INL} - \text{correlation window lines}}{\text{N - 1}}$$

3. Establish input (raw) space tie-point locations

   3.1 For j = 0 to N-2

$$y[j] = \frac{\text{correlation window lines}}{2} + j * \text{spacing y}$$

   3.2 $\text{y}[\text{N} - 1] = INL - \frac{\text{correlation window lines}}{2}$

   3.3 For i = 0 to M-2

$$\text{x}[i] = \frac{\text{correlation window samples}}{2} + i * \text{spacing x}$$

   3.4 $\text{x}[\text{M-1}] = INS - \frac{\text{correlation window samples}}{2}$

4. Project inputs space tie-points locations to output space

   4.1 For j=N-1

      4.1.1 For i=M-1
      Map input space tie-point location to output space using grid mapping coefficients.
         tie-point location y = $b_0 + b_1 * x[i] + b_2 * y[j] + b_3 * x[i] * y[j]$
         tie-point location x = $a_0 + a_1 * x[i] + a_2 * y[j] + a_3 * x[i] * y[j]$
      Where (See note #7):
      a_n = forward sample mapping coefficients for zero elevation plane retrieved from the TIRS LOS projection grid
      b_n = forward line mapping coefficients for zero elevation plane retrieved from the TIRS LOS projection grid

4.3.6.6.1.3 Stage 4 - Calculate Individual Point-by-Point Band Displacements

Normalized cross correlation is used to measure spatial differences between the reference and search windows extracted from the two bands being compared. The normalized cross correlation process helps to reduce any correlation artifacts that may arise from radiometric differences between the two image sources. The correlation process will only measure linear distortions over the windowed areas. By choosing

appropriate correlation windows that are well distributed throughout the imagery, non-linear differences between the image sources can be found. Normalized gray-scale correlation has the following formula:

$$R(x,y) = \frac{\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2}\left[\left(f(j,i)-\bar{f}\right)\left(g(x+j,y+i)-\bar{g}\right)\right]}{\left[\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2}\left(f(j,i)-\bar{f}\right)^2\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2}\left(g(x+j,y+i)-\bar{g}\right)^2\right]^{1/2}}$$

Where:

N = M = Correlation window size in lines and samples
R = correlation surface (N,M) (See note# 10)
f = reference window (N,M)
g = search window (N,M)

$$\bar{f} = \frac{1}{(M+1)(N+1)}\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2} f(j,i)$$

$$\bar{g} = \frac{1}{(M+1)(N+1)}\sum_{j=-N/2}^{N/2}\sum_{i=-M/2}^{M/2} g(x+j,y+i)$$

Normalized cross correlation will produce a discrete correlation surface (i.e., correlation values at integer x,y locations). A sub pixel location associated with the displacement is found by fitting a polynomial around a 3x3 area centered on the correlation peak. The polynomial coefficients can be used to solve for the peak or sub pixel location. Once the discrete correlation has been calculated and the peak value within these discrete values has been found the subpixel location can be calculated:

$$P(y,x) = a_0 + a_1x + a_2y + a_3xy + a_4x^2 + a_5y^2$$

Where
P(x,y) is polynomial peak fit
x = sample direction
y = line direction

Set up matrices for least squares fit of discrete R(x,y) to x/y locations.

or: [Y] = [X] [a]

Note that R(x,y) is relative to the peak, the total offset will need to have the integer line offset and sample offset added to the subpixel location to have the total measured offset. Solving for the peak polynomial using least squares:

$$[a] = ([X]^T[X])^{-1}[X]^T[Y]$$

Calculating the partial derivative of P(x,y) in both the x and y directions, setting the partial equations to zero, and solving the partials for x and y, gives the subpixel location within the subpixel 3x3 window.

$$\frac{\partial}{\partial x} P(x,y) = a_1 + a_3 y + 2a_4 x = 0$$
$$\frac{\partial}{\partial y} P(x,y) = a_2 + a_3 x + 2a_5 y = 0$$

Set partial equations equal to zero and solve for x and y:

$$\text{Subpixel x offset} = \frac{2a_1 a_5 - a_2 a_3}{a_3^2 - 4a_4 a_5}$$
$$\text{Subpixel y offset} = \frac{2a_2 a_4 - a_1 a_3}{a_3^2 - 4a_4 a_5}$$

The steps for mensuration, calculating the total offset measured, and how they fit in the overall procedure is given in the processing steps section.

See Ground Control Correlation Algorithm (Section 4.1.6) for prototype specifications of correlation processes.

4.3.6.6.1.4 Stage 5 - Removing Outliers Using the t-distribution

Once all the line and sample offsets have been measured and the first level of outlier rejection has been performed, a check against the maximum allowable offset and the minimum allowable correlation peak, the measurements are further reduced of outliers using a Student-t outlier rejection.

Given a t-distribution tolerance value, outliers are removed within the dataset until all values deemed as “non-outliers” or “valid” fall inside the confidence interval of a t-distribution. The tolerance, or associated confidence interval, is specified per run (or processing flow) and usually lies between 0.9-0.99. The default value is 0.95. The number of degrees of freedom of the dataset is equal to the number of valid data points minus one. The steps involved in this outlier procedure are given below. The process listed works on lines and samples simultaneously, calculating statistics independently for each.

Student-t Outlier Rejection Processing steps.
If a SCA-separated image is being analyzed, the outlier rejection sequence is executed independently on each SCA. For SCA-combined images, the outlier rejection logic is performed for all points at once.

For each SCA:

1. Calculate mean and standard deviation of data for lines and samples (see stage #6).

$$\text{mean offset} = \frac{1}{N}\sum_{i=0}^{N} \text{offset}_i$$

$$\text{standard deviation} = \frac{1}{N-1}\sum_{i=0}^{N} (\text{offset}_i - \text{mean offset})^2$$

Where:

N = number of valid offsets measured (above peak threshold and below maximum offset)

Two means and standard deviations are calculated, one for the line direction and one for the sample direction.

2. Find largest offset and compare it to outlier threshold.

   2.1. Calculate two-tailed t-distribution (T) value for current degree of freedom (N-1) and confidence level α.

   2.2. Calculate largest deviation from the mean allowable for the specified degree of freedom and α:
   $\Delta line = \sigma_{line} * T$
   $\Delta sample = \sigma_{sample} * T$
   Where:
   σ_{line} = standard deviation of valid line offsets
   σ_{sample} = standard deviation of valid sample offsets

   2.3. Find valid data point that is farthest from the mean.
   $\text{max line}_i = \text{MAX}\{ \text{line offset - mean line offset}\}$
   $\text{max sample}_j = \text{MAX}\{ \text{sample offset - mean sample offset}\}$
   Where:
   The maximum is found from all valid offsets
   i is the tie-point number of max line
   j is the tie-point number of max sample

   2.4. If valid data point that is farthest from the mean is greater than the allowable Δ then the valid point is flagged as outlier.
   if $\text{max line}_i > \Delta line$ or $\text{max sample}_j > \Delta sample$ then
   if($\text{max sample}_j / \sigma_{sample} > \text{max line}_i / \sigma_{line}$)
   tie-point j is marked as an outlier
   else
   tie-point i is marked as an outlier
   else no outliers found

3. Repeat 1 and 2 above until no outliers are found.

4.3.6.6.1.5 Stage 6 - Calculating Measured Statistics

The mean, standard deviation, minimum, maximum, median, and root-mean squared offset (RMS) are calculated from the tie-points that pass all outlier criteria; below maximum offset, above peak threshold, and student t-distribution test. The calculation for mean, standard deviation, and RMS are shown below where x_i is the measured offset.

mean: $m_x = \frac{\sum_{i=0}^{N-1} x_i}{N}$

standard deviation: $\sigma_x = \sqrt{\frac{\sum_{i=0}^{N-1} (x_i - m_x)^2}{N-1}}$

RMS: $RMS_x = \sqrt{\frac{\sum_{i=0}^{N-1} (x_i)^2}{N}}$

4.3.6.6.1.6 Band Accuracy Assessment Processing steps

Windows extracted from imagery have the user entered dimensions; correlation window lines and correlation window samples. Correlation parameters have been read or set as default values; maximum offset, fit method, correlation peak, fill data range, fill threshold. The bands should be indexed so that the TIRS 10.8 micrometer band is used as a reference to all other bands.

1. For SCA = Number of SCAs to process (1 if SCA-combined, 3 if SCA-separated)

    1.1. For rband = Total number of TIRS and OLI bands to process

        if rband is equal to OLI PAN use reduced PAN image file

    1.2. For sband = rband + 1 to Total number of TIRS and OLI bands to process

    1.3. For index = Number of tie-points to process

        1.3.1. Read current tie-point chip and tie-point location x,y
        Set tie-point flag to unsuccessful

        1.3.2. Extract sband window (of imagery) at tie-point location x,y

        1.3.3. Extract rband window (of imagery) at tie-point location x,y

        1.3.4. Count number of pixels in rband window that is within fill range.

count = 0
For i=0 to number of pixels in correlation window
    If rband pixel is > fill min and rband pixel is < fill max
        count++

1.3.5. Check number of rband pixels counted against fill threshold/percentage.

$$\text{if } \frac{\text{count}}{\text{correlation window size}} > \text{fill threshold}$$

        increment index to next tie-point location
    else
        continue

1.3.6. Count number of pixels in sband window that is within fill range.
count = 0
For i=0 to number of pixels in correlation window
    If sband pixel is > fill min and sand pixel is < fill max
        count++

1.3.7. Check number of sbands pixels counted against fill threshold/percentage.

$$\text{if } \frac{\text{count}}{\text{correlation window size}} > \text{fill threshold}$$

        increment index to next tie-point location
    else
        continue

1.3.8. Perform normalized gray-scaled correlation between rband and sband windowed images, calculating correlation surface *R* (See Stage 4 and notes #9 and #10).

1.3.9. Find peak within correlation surface
Max = *R*(0,0)
For i=0 to correlation window number of lines -1
    For j=0 to correlation window number of samples -1
        If *R*(i,j) > max then
            Max = *R*(i,j)
            line offset = i
            sample offset = j

1.3.10. Check correlation peak against threshold
if max > peak threshold
    continue
else
    set tie-point flag to outlier and choose next tie-point

1.3.11. Measure subpixel peak location (see stage #4)
    Δsub-line

Δsub-sample

1.3.12. Calculate total pixel offset
total line offset = line offset + Δsub-line
total sample offset = sample offset + Δsub-sample

1.3.13. Check offset against maximum displacement offset

$$\text{total displacement} = \sqrt{(\text{total line offset})^2 + (\text{total sample offset})^2}$$

if (total displacement > maximum displacement)
Set tie-point flag to outlier and choose next tie-point
Else
Set tie-point flag to valid

1.4. Store SCA and band combination (rband-to-sband) tie-point mensuration information, correlation success, and offsets measured. See Table 4-44.

2. For SCA = 1 to Number of SCAs to process

2.1. For band combination = 1 to Number of band combinations

2.1.1. Perform t-distribution outlier rejection (See stage #5).

2.2. Store SCA and band combination final individual tie-point information and outlier flag. See Table 4-45.

3. For SCA = 1 to Number of SCAs to process

3.1 For band combination = 1 to Number of band combinations

3.1.1. Calculate mean, minimum, maximum, median, standard deviation, and root mean squared offset.

3.1.2. Store SCA and band combination statistics. See Table 4-46.

4. Perform trending if trending flag is set to yes

4.1. Check results against trending metrics
For each band of each SCA
if measured Standard Deviation > trending metric
exit trending
If there are no Standard Deviation > trending metric perform trending

4.3.6.7 Output files

The output files listed below for the TIRS BRAA follow the philosophy of the Advanced Land Imagery Image Assessment System (ALIAS) Band-to-Band (B2B) Characterization output files in that they are made generic so that the same format can

be used elsewhere. Therefore, some fields like latitude, longitude, and elevation may not apply to the application and would be filled with zeros or nominal values.

All output files contain a standard header. This standard header is at the beginning of the file and contains the following:

1. Date and time the file was created
2. Spacecraft and instrument(s) pertaining to measurements.
3. Off-nadir (roll) angle of spacecraft/instrument.
4. Acquisition type
5. Report type (band-to-band)
6. Work order ID of process (left blank if not applicable)
7. WRS path/row
8. Software version that produced report.
9. L0R image file name

The data shown within Table 4-46 listed below is stored in the database. Database output will only be performed if the standard deviation statistics for every band and every SCA are below the thresholds (trending metrics) stored in the CPF (see note #3). The statistics stored per band per SCA will be used for trending analysis of the band registration accuracy of the TIRS instrument. Results produced through a time-series analysis of this data stored, over a set time interval or multiple image files, will determine if new LOS Legendre coefficients will need to be generated from the TIRS Band-to-Band Calibration Algorithm (See TIRS Band-to-Band Calibration Algorithm, Section 4.3.4). These statistics may also be needed for providing feedback to the L8/9 user community about the band registration of L8/9 products generated.

Field	Description
Date and time	Date (day of week, month, day of month, year) and time of file creation.
Spacecraft and instrument source	Landsat 8/9 and TIRS (and OLI if applicable)
Processing Center	EROS Data Center SVT
Work order ID	Work order ID associated with processing (blank if not applicable)
WRS path/row	WRS path and row (See note #11)
Software version	Software version used to create report
Off-nadir angle	Off-nadir roll angle of processed image file (See note #12)
Acquisition Type	Earth viewing (for TIRS)
L0R image file	L0R image file name used to create L1TP
Processed image file name	Name of L1TP used to create report
Reference bands	Reference bands used in band assessment
Search bands	Search bands used in band assessment
Heading for individual tie-points	One line of ASCII text defining individual tie-point fields.
For each tie-point:	
Tie point number	Tie-point index/number in total tie-point list
Reference line	Tie-point line location in reference image (band)
Reference sample	Tie-point sample location in reference image (band)

Field	Description
Reference latitude	Tie-point latitude location
Reference longitude	Tie-point longitude location
Reference elevation	Elevation of tie-point location (see note #13)
Search line	Tie-point line location in search image
Search sample	Tie-point sample location in search image
Delta line	Measured offset in line direction
Delta sample	Measured offset in sample direction
Outlier flag	1=Valid, 0=Outlier
Reference band	Reference band number
Search band	Search band number
Reference SCA	SCA number that reference window was extracted or 0 if an SCA-combined L1TP image was used
Search SCA	SCA number that search window was extracted or 0 if an SCA-combined L1TP image was used
Search image	Name of search image
Reference image	Name of reference image

Table 4-44. Band Registration Accuracy Assessment Data File

Field	Description
Date and time	Date (day of week, month, day of month, year) and time of file creation.
Spacecraft and instrument source	Landsat 8/9 and TIRS (and OLI if applicable)
Processing Center	EROS Data Center SVT
Work order ID	Work order ID associated with processing (blank if not applicable)
WRS path/row	WRS path and row (See note #11)
Software version	Software version used to create report
Off-nadir angle	Off-nadir pointing angle of processed image file (See note #12)
Acquisition Type	Earth viewing for TIRS
L0R image file	L0R image file name used to create L1TP
Processed image file name	Name of L1TP used to create report
Number of records	Total number of tie-points stored in file
Heading for individual tie-points	One line of ASCII text defining individual tie-point fields.
For each band combination	
Combination header	Number of points in combination, reference band number, search band number.
For each tie-point:	
Tie point number	Tie-point index/number in total tie-point list
Reference line	Tie-point line location in reference image (band)
Reference sample	Tie-point sample location in reference image (band)
Reference latitude	Tie-point latitude location
Reference longitude	Tie-point longitude location
Reference elevation	Elevation of tie-point location
Search line	Tie-point line location in search image
Search sample	Tie-point sample location in search image
Delta line	Measured offset in line direction
Delta sample	Measured offset in sample direction

Field	Description
Outlier flag	1=Valid, 0=Outlier
Reference band	Reference band number
Search band	Search band number
Reference SCA	SCA number that reference window was extracted from or 0 if an SCA-combined L1TP input image was used
Search SCA	SCA number that search window was extracted from or 0 if an SCA-combined L1TP input image was used
Search image	Name of search image
Reference image	Name of reference image

Table 4-45. Band Registration Accuracy Assessment Residuals File

Field	Description	Database Output
Date and time	Date (day of week, month, day of month, year) and time of file creation.	Yes
Spacecraft and instrument source	Landsat 8/9 and TIRS (and OLI if applicable)	Yes
Processing Center	EROS Data Center SVT	Yes
Work order ID	Work order ID associated with processing (blank if not applicable)	Yes
WRS path/row	WRS path and row (See note #12)	Yes
Software version	Software version used to create report	Yes
Off-nadir angle	Off-nadir pointing angle of processed image file (See note #13)	Yes
Acquisition Type	Earth viewing for TIRS	Yes
L0R image file	L0R image file name used to create L1TP	Yes
Processed image file name	Name of L1TP used to create report	No
t-distribution threshold	Threshold used in t-distribution outlier rejection	Yes
For each band combination of each SCA processed		
Reference band	Reference band of statistics	Yes
Search band	Search band of statistics	Yes
SCA	SCA number of statistics or 0 if SCA-combined L1TP input image was used	Yes
Total tie-points	Total number of tie-points for band combination of SCA	Yes
Correlated tie-points	Number of tie-points that successfully correlated for band combination of SCA	Yes
Valid tie-points	Total number of valid tie-points for band combination of SCA after all outlier rejection has been performed	Yes
For both line and sample direction:	All statistics are given in terms of pixels	
Minimum offset	Minimum offset within all valid offsets	Yes
Mean offset	Mean offset of all valid offsets	Yes
Maximum offset	Maximum offset within all valid offsets	Yes
Median offset	Median offset within all valid offsets	Yes
Standard deviation	Standard deviation of all valid offsets	Yes
Root-mean-squared	Root mean squared offset of all valid offsets	Yes

Table 4-46. Band Registration Accuracy Assessment Statistics Output File

4.3.6.7.1 Assessing Band Registration (Accessing Statistics Stored in Database)

The Band Accuracy Assessment statistics stored in the database will need to be accessed by the geometric CalVal team. Delineation, or essentially data base querying, will be done by the following or a combination of the following:

1. Date range of image acquisition or processing date
2. By SCA number (including 0 for SCA-combined results)
3. By band number
4. By acquisition type (Nadir, off-nadir, Lunar)
5. By geographic location of image extent.

At a minimum access to the Band Accuracy Assessment statistics is needed. Simple tools, such as an SQL queries, would be beneficial to the geometric CalVal team but are not absolutely necessary as they could be developed later through other means.

4.3.6.8 Notes

Some additional background assumptions and notes include the following:

1. Correlation parameters, minimum correlation peak and maximum offset, are stored and retrieved from the CPF.
2. Tools that analyze the query results will be needed to generate summary statistics: scene statistics, individual bands per SCA, SCA summary, band summary. These statistics would ultimately be provided to the user as summary statistics in an image quality assessment for the user community.
3. As shown in the input table, there will be metrics, based on calculated statistics, as to whether trending should be performed or not. This metric would be provided to avoid having garbage stored in the database. The metric values would be stored and retrieved from the CPF. There would be one metric per band per SCA which would be compared to the standard deviation statistics for each SCA. The criteria to check for trending are shown in Section 4.1 of the Band Accuracy Assessment Processing steps section.
4. Band Accuracy statistics stored within the database will be accessed for analysis.
    a. *Accessed according to a specific date range.*
    b. *Accessed according to a specific band or SCA.*
    c. *Accessed according to a specific geographic location.*
    d. *Accessed according to acquisition type (nadir, off-nadir, lunar).*

    This data accessed will be retrieved and stored within a comma delimited file. The methodology used to access the database could be an SQL script.
5. Data stored within the database will be accessed for time series analysis.
    a. Data would be pulled out by scene/SCA band pairs for a user-specified time period.
    b. Statistics over multiple scenes would be calculated per SCA and/or per band. Then combined them into the SCA and/or band average statistics.

c. Results could be compared to the band registration spec. These results could serve as triggers to other events, i.e., new CPF generation and testing.
d. Results could be used to verify conformance with product specifications.

These calculations could be performed within the methodology used to access the data from the database (SQL script).

6. Tie-point locations could also be stored and used as projection Y and X coordinates. The appropriate conversions must be done when converting between projection coordinates and line and sample locations when extracting image windows between bands. This transformation should also include any rotation due to path orientated projections.
7. The code uses a library call that maps any input point with a given elevation to output space. For BRAA, the elevation for the mapping point is set to zero. Since for BRAA the reference and search output space are the same, output line/sample in output reference space should be line/sample in output search space.
8. The c and d parallax coefficients are needed for each band or each SCA for every grid cell point. Therefore if the coefficients were stored as arrays stacked by grid column and then grid row for a particular input pixel that fell within grid cell column N and grid cell row M the c and d coefficients needed for that pixel would be indexed by: index = (M * number of grid columns + N) * 2. The factor of 2 is due to the fact the parallax odd/even effects are mapped as linear therefore 2 coefficients are stored for each the odd and even pixels of a grid cell.
9. The gray-scale correlation process, or surface, can be implemented using a Fast Fourier Transform (FFT).
10. The correlation surface could be smaller than the search window depending on the search area or maximum offset.
11. This tie point residual file structure is also used for the image registration accuracy characterization algorithm so it includes fields that are not required for both algorithms. An example is the elevation field, which is set to 0 for this algorithm.
12. The correlation result fit method defines the algorithm used to estimate the correlation peak location to subpixel accuracy. Only the quadratic surface fitting method described in this algorithm description is supported in the baseline algorithm. The least squares correlation technique is not used for TIRS band registration assessment. Since the source images are oversampled to 30 meter resolution, the robustness of the normalized gray-scale correlation is more important than the subpixel precision of the least squares method.

4.3.7 TIRS MTF Enhancement

4.3.7.1 Background/Introduction

The TIRS instrument exhibited poorer than intended spatial performance during thermal vacuum testing that resulted in a waiver being granted prior to the launch of Landsat 8. Though originally believed/hoped to have been largely due to ground support equipment limitations, the degraded performance was confirmed by on-orbit measurements, and was subsequently traced to the non-ideal behavior of the QWIP detectors. A

reconstruction filter, derived from the pre-launch model of TIRS spatial performance, can be used to improve the spatial response of the processed data. Unfortunately, this comes at some cost in the form of overshoot (ringing) and the unwanted enhancement of noise and residual radiometric artifacts such as detector-to-detector striping.

The details of the formulation of the reconstruction filter(s) are beyond the scope of this algorithm. For present purposes, the filter kernels will be taken as given inputs which are to be applied to the Level 1R (radiometrically corrected but geometrically raw) TIRS data. Nevertheless, in the interest of providing context, the reconstruction filters are created using the following procedure:

1. Construct a model of actual TIRS spatial performance in the form of a system transfer function. The functions derived during pre-launch thermal vacuum testing serve this purpose.
2. Construct a model of the nominal or desired TIRS spatial response, i.e., a nominal system transfer function that would just meet the spatial specifications.
3. In the frequency domain, divide the nominal transfer function by the actual transfer function out to the frequency at which the actual response goes below the system noise floor (estimated from the SNR), to yield the filter transfer function.
4. Window the resulting filter transfer function to suppress filter frequencies that are above the TIRS Nyquist frequency.
5. Transform the filter transfer function into the space domain to form a filter kernel.
6. Window the filter kernel to the desired size, and renormalize.

Due to variations in spatial performance between bands and even across the TIRS focal plane, separate filters were derived for each TIRS band and SCA. The kernels are all zero phase (i.e., symmetric) by design to avoid introducing any geometric shifts.

This algorithm describes the procedure whereby the TIRS MTFC kernels are applied to the TIRS Level 1R image data, just prior to image resampling. While conceptually a straightforward convolution operation, a slight complication is introduced by the Level 1R image geometry – specifically the presence of geometrically offset detector samples from the redundant detector row. The TIRS geometric model identifies those detectors taken from the redundant row as well as the magnitude of the row offset. Using this information, the convolution operator shifts the filter kernel weights in the line direction to compensate. Once the filter has been applied to the TIRS image samples within each SCA, the updated data are written back into the original L1R image file. This avoids the creation of another large file and reduces the impact to the image processing flow.

The spatial sharpening applied by this algorithm has been shown to improve TIRS/OLI image correlation accuracy. So, even if it is never used for standard TIRS L1TP product generation, this capability contributes to the TIRS Alignment Calibration and TIRS SSM Calibration procedures as an optional processing step.

4.3.7.2 Dependencies

The TIRS MTFC algorithm assumes that TIRS radiometric correction processing has created the Level 1R image file and that a LOS model has been created. Both the systematic LOS model and the precision LOS model contain the information needed by this algorithm, so either would work.

4.3.7.3 Inputs

The TIRS MTFC algorithm uses the inputs listed in the following table. Note that some of these “inputs” are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data).

Algorithm Inputs
ODL File (implementation)
CPF File Name
Level 1R File Name
LOS Model File Name (either Systematic or Precision LOS Model File will work)
Band List
CPF file contents (prototype implementation stores the MTFC kernels in the CPF)
MTFC Kernel Dimensions
MTFC Kernel Weights (one set per band/SCA)
Level 1R image file contents
Radiometrically corrected image samples organized by band and SCA
LOS Model file contents (see TIRS LOS Model Creation Algoriithm for details)
Detector offset/delay table (detector offsets for each detector in each band) – from CPF
Nominal detector alignment fill table
L0R detector alignment Fill Table

4.3.7.4 Outputs

Updated Level 1R Image File
Radiometrically corrected spatially sharpened image samples organized by band and SCA

4.3.7.5 Procedure

The TIRS MTF compensation algorithm applies an externally generated MTF compensation filter kernel to the input L1R TIRS data, updating the L1R data in place. This avoids the need to create a separate (large) new L1R output file in which only the TIRS bands have been changed. The filter kernels vary by band and SCA and are derived using models of the TIRS system transfer function. The details of the filter formulation are beyond the scope of this algorithm description so, for present purposes, the filter weights are simply an input loaded from the Calibration Parameter File. One important constraint on these filters is that they must be zero phase (i.e., symmetric) to avoid introducing geometric shifts into the L1R image data. In addition to the input/output L1R data and CPF filter weights, the algorithm uses the TIRS LOS model to account for the along-track shifts present in the L1R imagery due to the use of

deselected detectors (i.e., detectors from the secondary science row). The LOS model contains a table of the TIRS detector offsets by band and SCA.
The algorithm is quite simple, being essentially the convolution of a rectangular filter kernel with a rectangular image array. Only the handling of the detector deselect L1R image geometry provides a minor wrinkle. The algorithm procedure is as follows:

1. Load the LOS model structure from the input model file.
2. Load the CPF structure from the input file.
3. Load the MTF compensation kernels from the CPF (one per band/SCA).
4. Open the TIRS L1R file for read and write access.
5. For each TIRS band:
    a. Read the L1R image data (all SCAs).
    b. For each SCA:
        i. Get the filter kernel for the current band/SCA from the set loaded from the CPF.
        ii. Construct the detector offset table for this band/SCA using the information in the LOS model.

    Note: The most important contributor is the detector delay table for the current band/SCA, but the model also contains, and the algorithm uses, the L0R fill table copied from the L0R input file, and the nominal L0R fill offset. In practice the latter two values are always zero but they are included in the calculations for consistency with other algorithms (e.g., the resampler). Note that should a L1R dataset containing fill be encountered, the fill would corrupt the MTFC process near the top and bottom of the image. A warning message to this effect is included to protect against this eventuality. Logic to explicitly locate and avoid fill could be added, but would add unnecessary complexity to the indexing routines.

        For detector k:
        offset[k] = round(detector_delay[k] – l0r_fill[k] + nominal_fill)

        iii. Loop through the input image by image index. The Convolve One Image Point Sub-Algorithm described below, applies the MTFC filter at this image location. The filtered image values are stored in a separate buffer until the entire SCA has been filtered.
        iv. Overwrite the input image for this SCA with the filtered image.
    c. Write the filtered image band back into the original L1R image file.
6. Close the updated L1R file.

Convolve One Image Point Sub-Algorithm
The current image point is specified by its linear index, that is, the sequential location within the image buffer.

1. The index value is converted to an equivalent line/sample coordinate pair in ideal image space, accounting for deselected detector offsets and the image dimensions (nsamp, nline) using the image_line_sample unit:
    a. Compute the sample coordinate:
    sample = index % nsamp, where % indicates a remainder operation.
    b. Compute the line coordinate:

line = (index – sample) / nsamp + offset[sample].

2. Initialize the filtered image value by setting it to zero:
   filt_img[index] = 0
3. Set the filter offsets to one half the filter dimensions, discarding the fraction:
   s_off = filter_ns / 2
   l_off – filter_nl / 2
4. Loop on the filter dimensions:
   a. Compute the ideal image line/sample coordinate corresponding to the current filter line (index il) and sample (index is):
      i. img_line = line – l_off + il
      ii. img_samp = sample – s_off + is
   b. Convert the ideal image line/sample to an image buffer index, checking for locations that fall outside the image area, using the image_index unit:
      i. Initialize the index to -1 (invalid number).
      ii. Check the sample value against the image bounds:
          If img_samp < 0 or img_samp > nsamp-1 then return index.
      iii. Adjust the sample value for detector offset:
          adj_line = img_line – offset[img_samp]
      iv. Check the line value against the image bounds:
          If adj_line < 0 or adj_line > nline-1 then return index
      v. Calculate the image buffer index:
          index = adj_line * nsamp + img_samp
   c. If the returned index is invalid (i.e., < 0 or > nsamp*nline-1), set the filtered image value to the original image value for this line/sample location, and break out of the filter loops to proceed to the next image point.
      filt_img[index] = orig_img[index]
   d. If the returned index is valid, multiply the original image value at this index by the current filter weight, and add the result to the current filtered image value:
      filt_img[index] += orig_img[index] * filter[il*filter_ns+is]
   e. Continue until all filter weights have been applied or an input image location outside the image bounds causes the filter loop to be broken.
5. Return the final filtered image value.

Figure 4-79 shows how the detector offset adjustment to the line coordinate, described in step 4b above, controls the application of the filter kernel to the input image. The kernel labeled a) in the figure is in an area unaffected by deselected detectors, so the kernel is convolved with the image without adjustment. At location b) the kernel must reach outside its nominal footprint to adjust the convolution for the deselected detector.

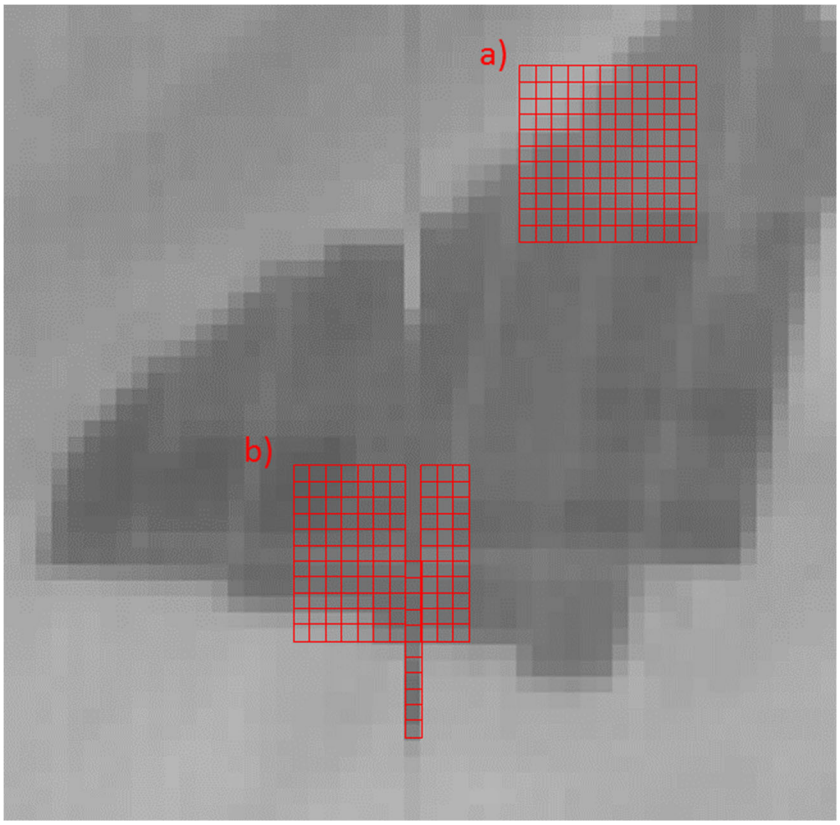

Figure 4-79. L1R Image Example Showing Deselected Detector Offset. a) Nominal filter convolution pattern. b) Convolution pattern accounting for deselect

Output Files

As noted above, the output of this algorithm is an updated version of the input L1R image file. The updates alter the values in some, but not all, of the image pixels in the TIRS thermal band image layers. Pixels that fall too close to the edge of one of the thermal band SCA images such that the MTFC sharpening kernel would have required data from outside the image to be fully applied, are left unaltered. The extent of the unaltered data varies with the MTFC filter dimensions and with the distribution and offset geometry of deselected detectors in each band/SCA. The updated data are written back into the original L1R HDF5 image file rather than creating a new copy. This facilitates integration of this processing step into the existing image resampling flow, and avoids the input/output and file storage burden of creating another large image file that is nearly identical (in the OLI reflective bands) to the original file.

4.3.7.6 Notes

Some additional background assumptions and notes include:

1. The initial intent of this algorithm is for use in geometric calibration process flows where the MTFC enhancement improves correlation accuracy between the TIRS and OLI spectral bands. Implementing this processing step in routine product generation would require further study of its impact on science, and the approval of the Landsat Science Team.

2. In light of item #1 above, locating the MTFC filter coefficients in the CPF, as was done in the prototype implementation, may not be the best solution. It was done this way for several reasons:
    a. It facilitates adoption of this capability as part of standard Level 1 processing. This could be particularly important if TIRS-2 should exhibit spatial performance that is worse (or more objectionable due to the lack of stray light) than TIRS-1.
    b. It avoided creating another input “system table” file that would have to be kept track of and managed.
    c. The MTFC parameters are CPF-like in the sense that they are sensor and mission specific.
    d. The parameters could change as knowledge of TIRS spatial performance improves (e.g., as a result of the ongoing QWIP detector characterization efforts) and the CPF management process accommodates parameter updates.
    e. It seemed like the easiest way to do it given the extensive existing CPF/ODL input infrastructure.
3. A disadvantage of using the CPF as the vehicle for the TIRS MTFC parameters is that it required modifications to IAS library routines.
4. The use of an automated test script for algorithm verification is an experiment to see if this approach simplifies algorithm delivery and verification.
5. This algorithm will not work on lunar data since L1R detector-to-detector alignment does not reflect the usual detector offsets in those datasets due to the reduced object space scan rate.

4.3.8 TIRS Scene Select Mechanism (SSM) Model Fit Algorithm

4.3.8.1 Background/Introduction

In December 2014, failures in the Landsat 8 TIRS SSM mirror encoder circuitry made it necessary to operate the instrument with the SSM in an open loop control mode (mode-0) with the SSM position encoder turned off. This anomaly was eventually resolved by switching to the redundant “B-side” TIRS electronics, allowing TIRS SSM operations to continue in the nominal closed loop control mode (mode-4). Subsequent anomalous behavior in the B-side electronics raised the possibility of having to perform sustained TIRS operations in mode-0, leading to a requirement for an operational SSM position estimation, modeling, and prediction capability. This capability would replace the SSM encoder position measurements provided in the downlinked ancillary data; which are unavailable when operating in the open loop mode-0 with the encoder turned off; with SSM position estimates calculated using a time-varying model of SSM position. This algorithm uses estimates of the SSM position, derived from individual calibration scenes, and any available SSM encoder telemetry to construct such a time-varying model of SSM motion/position. This model is then used to predict SSM pointing in support of TIRS Level 1 data processing.

The SSM calibration algorithm uses the trended output of the TIRS SSM calibration algorithm and combines those single-scene estimates of SSM position with available SSM encoder measurements collected during the first few minutes following a mode-4

to mode-0 switch, to construct an integrated measurement dataset (see note #1). These measurements are used to adjust the parameter values in a linear with exponential decay model that was empirically determined to track actual SSM motion following a mode-0 switch. The time density and duration of the measurement dataset determines which model parameters are candidates for adjustment in any given invocation of the model fit logic. Once a refined set of model parameter values are computed, they are used to generate a table of estimated SSM positions for the time period following the mode switch, suitable for use in the geometric correction of TIRS data acquired during that time period (see note #2).

The algorithm consists of several sub-functions, controlled by a graphical user interface (GUI), that together implement the steps needed to create the SSM position estimate records used in Level 1 data processing. These sub-functions include:

1) Identifying a new mode-4 to mode-0 switch event and creating an initial SSM position model for the time period following the event.
2) Modifying the parameters (i.e., start date/time) of an existing mode-0 switch event. This may be necessary to correct an event that was initially created based upon a planned switch time to the actual event time.
3) Gathering the available measurements of SSM position for the time period following the switch event, including both SSM encoder measurements reported in the TIRS telemetry and scene-based position measurements derived by the SSM calibration algorithm, and using these measurement data to solve for updates to the current SSM position model parameters. This includes using the time distribution of data points to determine the subset of model parameters that can be reliably solved for given the available data coverage.
4) Using the updated model parameter values to generate a table of SSM position estimates covering the time span following the corresponding mode-4 to mode-0 switch event.

Mode-4 to mode-0 switch events and the associated estimated SSM pointing model parameter values are stored in the geometric trending database so that they are available to serve as either the starting point of a new model fit operation using newly acquired data, or to generate an updated SSM position table to support Level 1 processing.

4.3.8.2 Dependencies

The TIRS SSM model fit algorithm assumes that either Level 1 processing or TIRS stored state of health (SSoH) data ingest processing has extracted, preprocessed, and stored (in the geometric trending data base) the available SSM encoder telemetry for the mode-0 operating period. It also assumes that the TIRS SSM Calibration Algorithm 6.3.8 has analyzed one or more scenes and created estimated SSM positions which have also been stored in the geometric trending data base.

4.3.8.3 Inputs

The TIRS SSM calibration algorithm uses the inputs listed in the following table. Note that some of these "inputs" are implementation conveniences (e.g., using an ODL parameter file to convey the values of and pointers to the input data). The second column in the table shows which algorithm operation: 1) mode-switch event identification (Add Event), 2) mode-switch event update (Edit Event), 3) data gathering and model fit (Fit Model), or 4) SSM position table generation (Generate Position); uses each input.

Algorithm Inputs	Operation
User Input	Control
Operation selection:	Control
1. Add new switch event.	Control
2. Edit existing switch event.	Control
3. Fit SSM position model.	Control
4. Generate SSM position table.	Control
Event date as year, day of year, second of day or UTC year, month, day, hour, minute, second.	Add Event Edit Event
Event selection (from displayed list)	Edit Event Fit Model
Fit Solution Acceptance	Fit Model
Position Table Start date/time (year, day of year, second of day)	Generate Position
Position Table End date/time (year, day of year, second of day)	Generate Position
Position Table Acceptance	Generate Position
CPF Service	Add Event Fit Model
Calibration Parameter File (CPF) Path	Add Event Fit Model
Calibration Parameter File (see note #3)	Fit Model Generate Position
SSM nadir reference angles (side A and side B)	Fit Model
Leap second table (for spacecraft clock to/from UTC conversion)	Fit Model
Default Fit Quality Threshold (new parameter – 1 value)	Fit Model
Telemetry and Image Observation Weights (new parameters – 2 values, one for telemetry observations and one for image observations)	Fit Model
Apriori Parameter Weights (new parameters – 8 values, one per model parameter)	Fit Model
Model Parameterization Break Points (new parameters – 4 values that define the time break points between the 1, 3, 5, 7, and 8 parameter versions of the model fit)	Fit Model
Model Observation Gap Windows (new parameters – 3 pairs of values, each pair defining a time window which, lacking any observations, would cause the introduction of apriori parameter observations)	Fit Model
Convergence Threshold Weights (new parameters – 8 values, one per model parameter)	Fit Model
Sampling Interval Table (new parameters – variable number, nominally 14, of pairs of values, each pair defining a start time, in	Generate Position

Algorithm Inputs	Operation
seconds, and a time increment, in seconds, to use in constructing the SSM position table entry times).	
SSM Mode Switch Event Table	All
Mode switch ID, year, day of year, seconds of day (UTC) (see Table 4-47)	All
TIRS_TELEMETRY_COMMAND Table	Fit Model
L0R_TIME_DAYS_FROM_J2000	Fit Model
L0R_TIME_SEC_OF_DAY	Fit Model
MC_ENCODER_FLAGS	Fit Model
SSM_MECH_MODE	Fit Model
SSM_ENCODER_POSITION_SAMPLE_2	Fit Model
TIRS_TELEMETRY_CIRCUIT Table	Fit Model
L0R_TIME_DAYS_FROM_J2000	Fit Model
L0R_TIME_SEC_OF_DAY	Fit Model
ELEC_ENABLED_FLAGS	Fit Model
TIRS_SSM_ESTIMATION Table (TIRS SSM Calibration Output)	Fit Model
SSM Angle Estimation Epoch as year, day of year, seconds of day	Fit Model
SSM Estimated Position in counts (or radians)	Fit Model
TIRS_SSM_ESTIMATION_SCA Table (TIRS SSM Calibration Output)	Fit Model
SCA Number	Fit Model
Post-Fit Along-Track RMSE	Fit Model
Post-Fit Across-Track RMSE	Fit Model
Number of Points	Fit Model
SSM Mode 0 Model Parameter Table	Add Event Fit Model Generate Position
Model ID, switch ID, algorithm version, nadir reference, model parameter values, date added, date disabled (see Table 4-48)	Add Event Fit Model Generate Position

4.3.8.4 Outputs

Algorithm Outputs	Operation
Output To User	All
Switch Event (switch ID, year, day of year, seconds of day (UTC)	Edit Event
Switch Event List	All
Invalid Event Message	Add Event Edit Event
Model Parameters (see Model Parameter Table contents above)	Fit Model
Model Fit Statistics (fit RMSE for telemetry observations, image observations, and all observations)	Fit Model
Updated Observations with Fit Residuals (observation date/time, seconds from mode switch, days from mode switch, observation type, measured encoder counts, modeled encoder counts, residual in counts)	Fit Model
Plot of observed and modeled encoder positions vs. time since switch (optional)	Fit Model
SSM Position Estimates (year, day of year, second of day, position in encoder counts)	Generate Position
CPF Service	Add Event

Algorithm Outputs	Operation
	Fit Model
CPF Request (based upon date)	Add Event Fit Model
SSM Mode Switch Event Table (See Table 4-47)	Add Event Edit Event
Switch event ID, year, day of year, seconds of day	Add Event Edit Event
SSM Mode 0 Model Parameter Table (See Table 4-48)	Add Event Fit Model
Model ID, switch ID, algorithm version, nadir reference, model parameter values, date added, date disabled	Add Event Fit Model
SSM Mode 0 Position Estimate Table (See Table 4-49)	Generate Position
Year, day of year, seconds of day, encoder position, quality flag, associated model ID, date added, date disabled.	Generate Position
SSM Model Fit Report File (see Table 4-51)	Fit Model
Report generation date/time and location	Fit Model
Mission (Landsat 8) and sensor (TIRS)	Fit Model
Original model parameters	Fit Model
Updated model parameters	Fit Model
Fit statistics	Fit Model
Updated observations with model fit residuals	Fit Model

4.3.8.5 Options

User Operation Selection (Add Event, Edit Event, Fit Model, Generate Positions)
TIRS SSM Model Parameter Trending User Confirmation
TIRS SSM Position Estimate Trending User Confirmation

4.3.8.6 Procedure

The TIRS SSM Model Fit Algorithm is used for on-orbit estimation of the TIRS scene select mechanism position time history while operating in open loop (mode-0) control. This algorithm is necessary to provide the SSM position information needed to process TIRS data collected in mode 0 with sufficient accuracy to meet the TIRS-to-OLI band registration, and the TIRS image registration, geodetic accuracy, and geometric accuracy requirements.

Mathematical Background
During the TIRS SSM anomaly of December 2014 through March 2015 it was determined experimentally that when released from closed loop control, the SSM reacts to the residual magnetic torque in the motor by moving, rapidly at first, and then at a decaying rate, away from the nominal nadir pointing position. It was found that the magnitude and variability of this post-switch motion can be reduced by applying decreasing-sized motor motions in alternate directions prior to releasing the SSM. This so-called “pendulum” maneuver is the planned operational implementation of mode 4 (closed loop) to mode 0 (open loop) switch operations if and when those become necessary to extend mission life.

An empirical model was fitted to the measured SSM positions following mode 0 switching events to allow predictions of SSM position for future times and to

smooth/regularize the image-based measurements used to monitor SSM position in the absence of SSM encoder data. This model includes an initial position offset, a position rate/slope parameter, and three decaying exponential terms with time constants varying from a few minutes to several days. The form of the model is:

$$P(t) = a_0 + a_1\left(1 - e^{-t_{seconds}/\tau_1}\right) + a_2\left(1 - e^{-t_{days}/\tau_2}\right) + a_3\left(1 - e^{-t_{days}/\tau_3}\right) + St_{days} \quad (1)$$

Where:

P(t) = SSM position offset from nominal nadir position (in counts) as a function of time from switch to mode 0 (t). The time is shown in units of both seconds and days to simplify the presentation of the equation by hiding the conversion factors.

a_0 = Initial constant offset parameter (in counts).

a_1 = Magnitude of the first (short time constant) exponential decay term (in counts).

a_2 = Magnitude of the second (medium time constant) exponential decay term (in counts).

a_3 = Magnitude of the third (long time constant) exponential decay term (in counts).

τ_1 = Time constant (in seconds) of the first exponential decay term.

τ_2 = Time constant (in days) of the second exponential decay term.

τ_3 = Time constant (in days) of the third exponential decay term.

S = Slope (long term rate) of SSM motion (in counts per day).

The parameters of this non-linear equation are solved for using measurements of SSM position derived from direct encoder telemetry (where available immediately after the switch) and from calibration scenes processed through the TIRS Scene Select Mechanism (SSM) Calibration Algorithm (refer to the algorithm description document of the same name for details) to generate scene-average SSM position estimates. The current best estimates of the parameter values are used as the starting point for a Taylor series expansion linearization of equation (1). This linearization requires the partial derivatives of the P(t) equation with respect to each of the eight model parameters:

$$\hat{P}(t,\theta_i+\Delta\theta) \cong \hat{P}(t,\theta_i)+\frac{\partial P}{\partial a_0}\Delta a_0+\frac{\partial P}{\partial a_1}\Delta a_1+\frac{\partial P}{\partial a_2}\Delta a_2+\frac{\partial P}{\partial a_3}\Delta a_3 + \frac{\partial P}{\partial \tau_1}\Delta\tau_1+\frac{\partial P}{\partial \tau_2}\Delta\tau_2+\frac{\partial P}{\partial \tau_3}\Delta\tau_3+\frac{\partial P}{\partial S}\Delta S \quad (2)$$

Where:

$$\theta_i = [a_{0i} \quad a_{1i} \quad a_{2i} \quad a_{3i} \quad \tau_{1i} \quad \tau_{2i} \quad \tau_{3i} \quad S_i]^T \quad (3)$$

= The vector of parameter values after the ith iteration.

$$\Delta\theta = [\Delta a_0 \quad \Delta a_1 \quad \Delta a_2 \quad \Delta a_3 \quad \Delta\tau_1 \quad \Delta\tau_2 \quad \Delta\tau_3 \quad \Delta S]^T \quad (4)$$

= The vector of parameter value corrections for the current iteration.

$$\frac{\partial P}{\partial a_0} = 1 \quad (5a)$$

$$\frac{\partial P}{\partial a_1} = \left(1 - e^{-t_{seconds}/\tau_1}\right) \quad (5b)$$

$$\frac{\partial P}{\partial a_2} = \left(1 - e^{-t_{days}/\tau_2}\right) \quad (5c)$$

$$\frac{\partial P}{\partial a_3} = \left(1 - e^{-t_{days}/\tau_3}\right) \quad (5d)$$

$$\frac{\partial P}{\partial \tau_1} = \frac{-a_1\, t_{seconds}}{\tau_1^2}\, e^{-t_{seconds}/\tau_1} \quad (5e)$$

$$\frac{\partial P}{\partial \tau_2} = \frac{-a_2\, t_{days}}{\tau_2^2}\, e^{-t_{days}/\tau_2} \quad (5f)$$

$$\frac{\partial P}{\partial \tau_3} = \frac{-a_3\, t_{days}}{\tau_3^2}\, e^{-t_{days}/\tau_3} \quad (5g)$$

$$\frac{\partial P}{\partial S} = t_{days} \quad (5h)$$

= The partial derivatives of the model equation with respect to the 8 parameters.

For each observation, the vector of partial derivatives, evaluated at the current parameter values and the time of the observation, provide the linearized observation coefficients:

$$A_j = \left[\frac{\partial P}{\partial a_0} \quad \frac{\partial P}{\partial a_1} \quad \frac{\partial P}{\partial a_2} \quad \frac{\partial P}{\partial a_3} \quad \frac{\partial P}{\partial \tau_1} \quad \frac{\partial P}{\partial \tau_2} \quad \frac{\partial P}{\partial \tau_3} \quad \frac{\partial P}{\partial S}\right]^T \quad (6)$$

= The vector of partial derivatives for observation j.

The linearized measurement, B_j, is the difference between the measured value of SSM position, P_j, and the value predicted by the model using the current model parameter values:

$$B_j = \left[P_j - \hat{P}(t_j, \theta_i)\right] \quad (7)$$

= The difference between the measured position for observation j and the current position model evaluated at the current parameter values and the jth observation time.

Each observation gets a weight based upon its source, encoder telemetry or calibration scene estimate, since the actual encoder observations are more precise than the image-based measurements. In each case the weight is the inverse of the estimated variance of the measurement.

$$W_j = \left[w_{encoder} \ or \ w_{image}\right] \quad (8)$$

The linearized coefficients, measurements, and weights for each observation are used to assemble the normal equations for a weighted least squares solution for the parameter correction vector:

$$N = \sum_j A_j \, W_j \, A_j^T \quad (9a)$$

$$C = \sum_j A_j \, W_j \, B_j \quad (9b)$$

$$\Delta\theta = N^{-1} \, C \quad (9c)$$

$$\theta_{i+1} = \theta_i + \Delta\theta \quad (9d)$$

With this formulation, individual parameters can be removed from the solution by manipulating the normal equations directly. Specifically, by zeroing out the column (in N) and row (in N and C) associated with the parameter to be deleted and inserting a value of 1 in the diagonal element of N corresponding to the deleted observation. This forces the correction for that parameter to be zero.

A priori pseudo-observations can also be injected to limit, but not fully constrain, the adjustment allowed for a particular, poorly observed, parameter. This is accomplished by adding a weight term to the diagonal element of N corresponding to the parameter, and adding the difference between the initial value of the parameter and the current estimated value of the parameter, multiplied by the weight term, to the element of C corresponding to the parameter. The weight term should be equal to the inverse of the estimated variance of the initial parameter value (i.e., $1/\sigma^2$). For example, to add an a priori observation to parameter i, set $N(i,i) = N(i,i) + 1/\sigma^2$ and $C(i) = C(i) + (\theta_0(i) - \theta(i))/\sigma^2$ where $\theta_0(i)$ is the initial value of parameter i and $\theta(i)$ is the current (last iteration) value of parameter i.

The iterative solution continues until it converges, i.e., the calculated corrections are acceptably small. The final set of parameter values can then be used to calculate the final observation residuals using equation (7) above.

Figure 4-80, Figure 4-81, and Figure 4-82 show the top level process flows for the overall SSM model fit algorithm, not just the least squares fit to the encoder and measured data. Add Event, Fit Model, and Generate Positions operations are available through a set of IAS utilities that help create the end-to-end steps needed in order to perform the calibration of the SSM's behavior. These diagrams are intended to show the high-level process flow and should not be construed as comprehensive data flow diagrams.

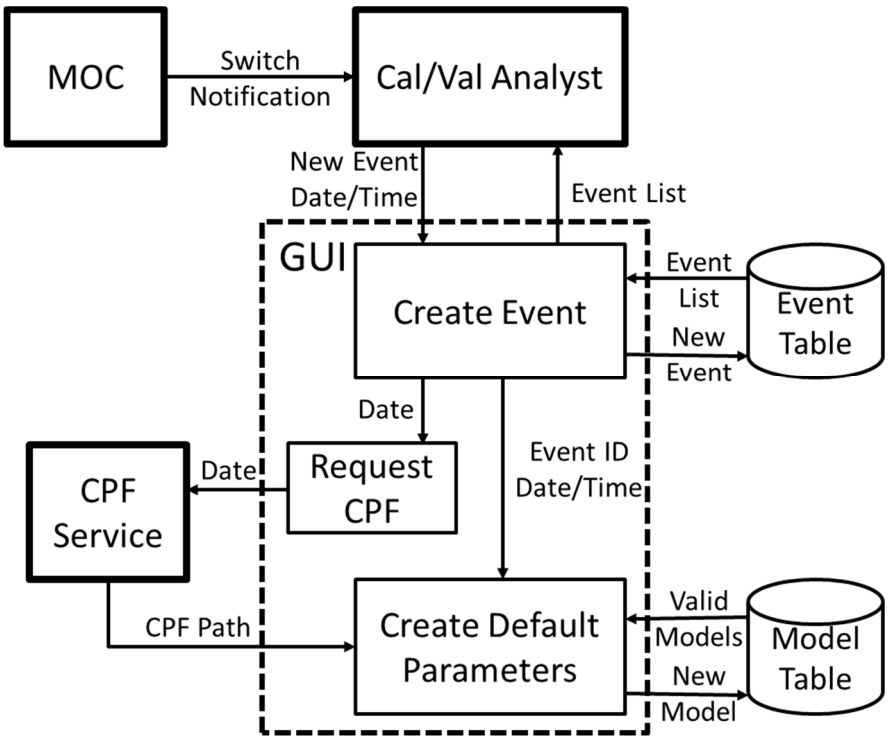

Figure 4-80. Add SSM Mode Switch Event Process Flow

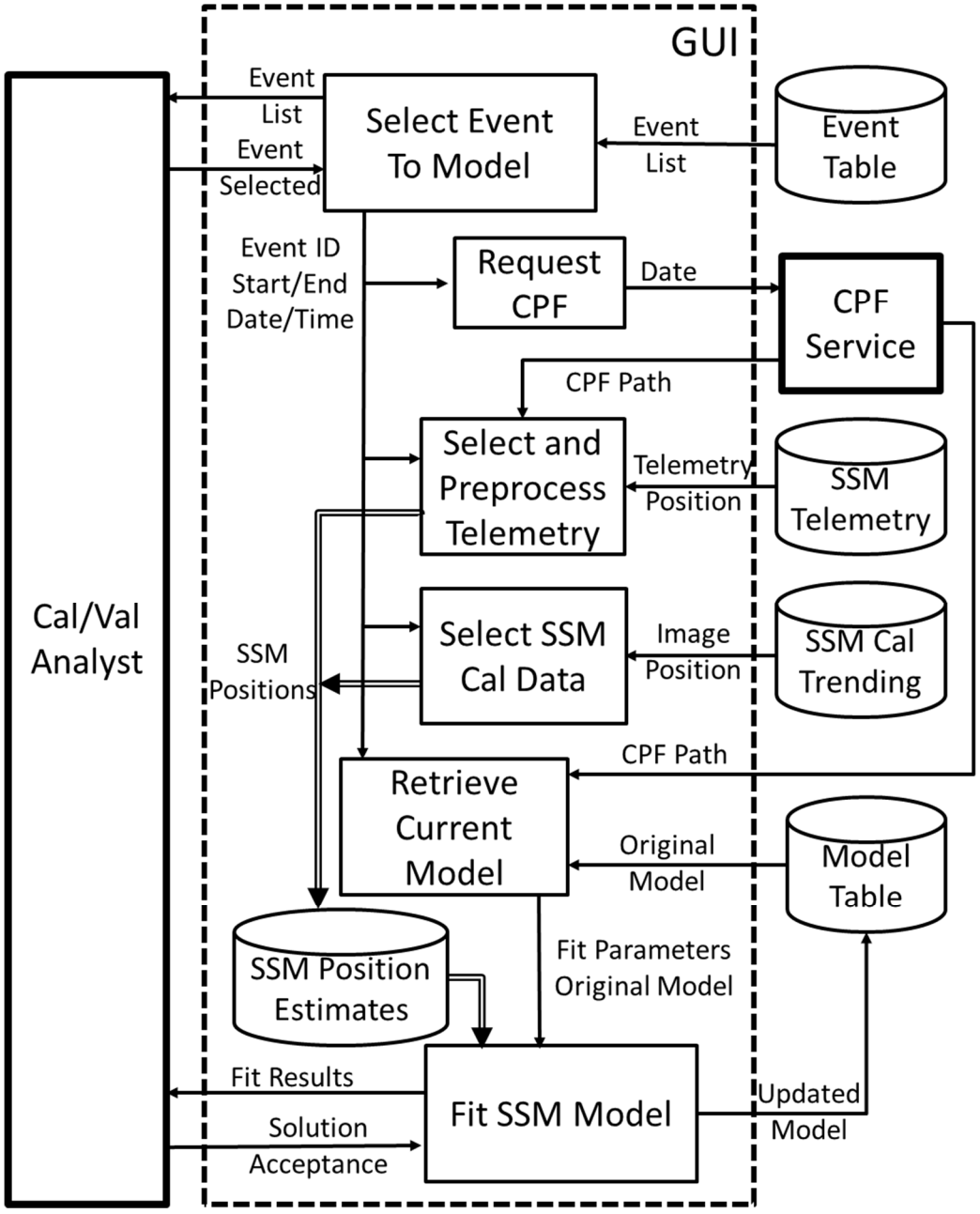

Figure 4-81. Fit SSM Model Process Flow

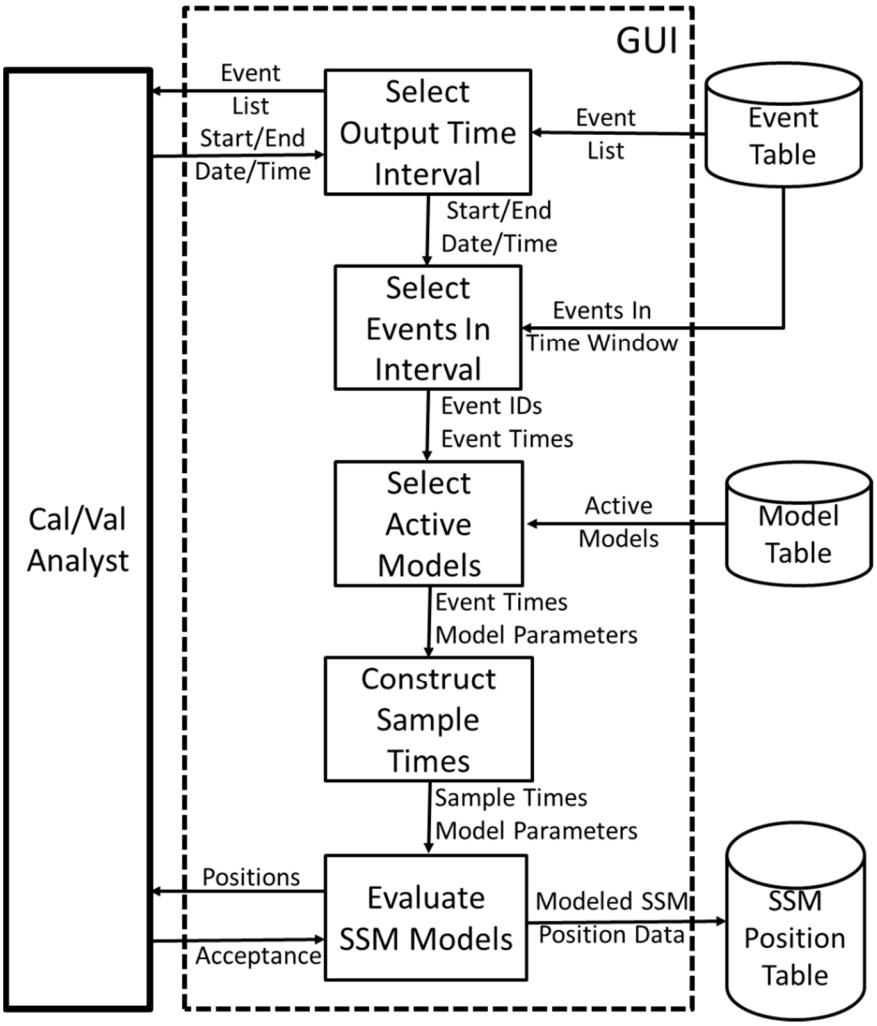

Figure 4-82. Generate SSM Position Table Process Flow

4.3.8.7 Algorithm Output Details

The TIRS SSM model fit algorithm populates three new database tables: an SSM mode switch event table (defined in Table 4-47) that uniquely identifies each mode switch event and stores the associated event date and time; an SSM mode 0 model parameter table (defined in Table 4-48) that stores the model parameter values needed to generate SSM position estimates as a function of time; and an SSM mode 0 position table (defined in Table 4-49) containing the actual SSM position estimates spanning the period of mode 0 operations, generated using the model parameters.

SSM Mode Switch Event Table	Units	Field Type
Switch event ID (key)	-	Sequence
Event year	Years	Integer
Event day of year	Days	Integer
Event seconds of day (UTC)	Seconds	Integer

Table 4-47. TIRS SSM Mode Switch Event Table Contents

The switch event ID field is suggested as an Oracle sequence but any mechanism for yielding uniquely identifiable records that can be easily referenced from other tables is acceptable.

SSM Mode 0 Model Parameters Table	Units	Field Type
Model ID (key)	-	Sequence
Associated mode switch ID (link to SSM Mode Switch Event Table)	-	External Key
Algorithm version number	-	Integer
Encoder Nadir Position	Counts	Integer
Constant Offset from Nadir (a_0)	Counts	Double
First Exponential Magnitude (a_1)	Counts	Double
Second Exponential Magnitude (a_2)	Counts	Double
Third Exponential Magnitude (a_3)	Counts	Double
First Exponential Time Constant (τ_1)	Seconds	Double
Second Exponential Time Constant (τ_2)	Days	Double
Third Exponential Time Constant (τ_3)	Days	Double
Long Term Slope (S) (in counts per day)	Counts per Day	Double
Date Added (date and time to nearest second)	Seconds	Oracle Date
Date Disabled (NULL or date and time to nearest second)	Seconds	Oracle Date

Table 4-48. TIRS SSM Mode 0 Model Parameter Table Contents

SSM Mode 0 Position Estimate Table	Units	Field Type
Year	Years	Integer
Day of year	Days	Integer
Seconds of day (UTC)	Seconds	Integer
Encoder position	Counts	Integer
Quality flag: 0 = predicted (i.e., generated before the event) 1 = updated/preliminary (i.e., generated during the event) 2 = final (i.e., generated after the event)	-	Integer
Associated model ID (link to SSM Mode 0 Model Parameters Table)	-	External Key
Date Added (date and time to nearest second)	Seconds	Oracle Date
Date Disabled (NULL or date and time to nearest second)	Seconds	Oracle Date

Table 4-49. TIRS SSM Mode 0 Position Estimate Table Contents

The contents of the SSM mode 0 position estimate table can be written to one or more ASCII text file(s) for dissemination to external users (e.g., International Cooperators). Only the first five fields (year, day of year, seconds of day, encoder position, and quality flag) would be included in the file output with regards to the SSM position. The header to this file contains the format version, creation date, start date, end date, and number of records. The contents for this file is shown in Table 4-50.

SSM Mode 0 Position Estimate Table ASCII Format	Units	Field Type
Format Version		Integer
Creation Date	YYYY:DOY:SOD UTC	String
Start Date	YYYY:DOY:SOD UTC	
End Date	YYYY:DOY:SOD UTC	
Number Records		Integer
For each SSM Mirror Record		Integer
Year	Years	Integer
Day of year	Days	Integer
Seconds of day (UTC)	Seconds	Integer
Encoder position	Counts	Integer
Quality flag: 0 = predicted (i.e., generated before the event) 1 = updated/preliminary (i.e., generated during the event) 2 = final (i.e., generated after the event)	-	Integer

Table 4-50. TIRS SSM Mode 0 Position Estimate Table Contents ASCII Version

The contents of the output TIRS SSM model fit report file are summarized in Table 4-51. Note that the first four fields listed are part of the standard report header. Most of this information (other than the standard report header) is presented to the user prior to acceptance of the model fit as valid.

Field	Description
Date and time	Date (day of week, month, day of month, year) and time of file creation.
Spacecraft and instrument source	L8 and TIRS
Processing center	EROS
Software version	Software version used to create report
Associated mode switch event ID	Mode switch event sequence number
Mode switch date and time	Year, day of year, seconds of day (UTC)
Original model ID	ID of model used as starting point
Algorithm version	Version number of algorithm used to create original model
Encoder nadir	Nadir reference value (in counts)
Original a_0	Original value of constant offset from nadir (in counts)
Original a_1	Original value of first exponential magnitude (in counts)
Original a_2	Original value of second exponential magnitude (in counts)
Original a_3	Original value of third exponential magnitude (in counts)
Original τ_1	Original value of first exponential time constant (in seconds)
Original τ_2	Original value of second exponential time constant (in days)
Original τ_3	Original value of third exponential time constant (in days)
Original slope	Original value of long term slope (in counts per day)
Original model date	Date/time original model was created
Updated model ID	ID of newly created model
Algorithm version	Version number of algorithm used to create updated model
Encoder nadir	Nadir reference value (in counts)

Field	Description
Updated a_0	Updated value of constant offset from nadir (in counts)
Updated a_1	Updated value of first exponential magnitude (in counts)
Updated a_2	Updated value of second exponential magnitude (in counts)
Updated a_3	Updated value of third exponential magnitude (in counts)
Updated τ_1	Updated value of first exponential time constant (in seconds)
Updated τ_2	Updated value of second exponential time constant (in days)
Updated τ_3	Updated value of third exponential time constant (in days)
Updated slope	Updated value of long term slope (in counts per day)
Updated model date	Date/time updated model was created
Telemetry RMSE	RMS fit residual for telemetry observations
Image RMSE	RMS fit residual for image observations
Combined RMSE	RMS fit residual for all observations
Telemetry count	Number of telemetry observations
Image count	Number of image observations
Total count	Total number of observations
Observation Records:	For each observation used in the fit:
Observation date/time	Year, day of year, seconds of day
Seconds from mode switch	Observation time offset, in seconds, from mode switch event
Days from mode switch	Observation time offset, in days, from mode switch event
Observation type	Type of observation: telemetry or image
Measured position	Measured position in encoder counts
Modeled position	Final modeled position in encoder counts
Position residual	Position residual (measured minus modeled) in encoder counts

Table 4-51. TIRS SSM Model Fit Report Details

4.3.8.8 Notes

Some assumptions, limitations, and implementation notes include:

1. The model fitting procedure is designed to work with data created as a result of executing nominal alternate SSM operations concept mode switch procedures. This implies that: 1) the mode switch will follow a “pendulum” maneuver to reduce the magnitude of subsequent SSM motion; 2) a reasonable amount (~2000 seconds, minimum) of encoder data will be captured following the mode switch; 3) the SSM will remain in mode 0 long enough to collect sufficient data to model its trajectory, nominally for 14 days but for at least 7 days; and 4) no extreme off-nadir spacecraft maneuvers (i.e., lunar calibrations) occur during the mode 0 period. Mode switch events that do not conform to these conditions are not expected to achieve acceptable model fits using this algorithm. This excludes most of the mode switch events during the initial SSM anomaly period from December 2014 to March 2015.
2. As a consequence of note #1, above, the SSM position table must be preloaded with externally generated SSM position records for mode 0 time periods that cannot be successfully modeled with the current algorithm. This applies to the original side A SSM anomaly time period.
3. The parameters identified as new CPF parameters could also be stored in an algorithm-specific configuration file, since they have no relevance for external users of Landsat 8 data.

4.4 Common Radiometry Algorithms

4.4.1 Dropped Frame Characterization

4.4.1.1 Background/Introduction

All dropped “frames” (aka dropped “lines” for this algorithm description), need to be accounted for during Ingest System (IS) processing and Product Generation System (PGS) processing.

Dropped frames are typically expected to occur onboard the spacecraft. The instrument will create a CRC value at the end of each video line. Once the data is on the ground, ingest will perform a CRC check on these values. Any check failure will result in the dropped frame being filled with zeros, and the setting of the CRC flag to fail in the OLI line header dataset.

Dropped frame conditions may also occur if a file within a given interval is missing. Such intervals will be archived for later processing. If or when selected, the interval may be broken up into two “sub-intervals” or the data may be filled to complete the interval. In the latter case, ingest will flag these all these frames as zero filled in the OLI line header dataset.

This algorithm will check the Line Header data in the ancillary file for all intervals and output any dropped frames to a Labeled Mask (LM) aka Artifact Mask (AM) and database.

4.4.1.2 Inputs

Description	Level	Source	Type
Line Header data	interval	Ancillary Data File	

4.4.1.3 Outputs

Description	Level	Target	Type
Dropped Frames	N_{band} x N_{frame}	AM	Int
# of Dropped Frames	N_{band} x N_{frame}	Db	Int

4.4.1.4 Procedure

1. Read the interval ancillary data file and extract the LineHeader Dataset
2. Verify CRC check and line fill information from the Line Status field.

```
{
For (band=0,band=9,band++)

Num_filllines = 0          ; /*line counter initialization*/
Int Linearray[9,Num_fillines] ;  /* array containing CRC  check*/
Int Filltotal[9];  /*array containing # of dropped frames per band*/
{
```

```
For  (line = 0, line=intervalsize, line++)
        {
       If (bit8 == 1)              ;  /*CRC check successful*/
            {
            If  (bit2 ==  1)          ; /*line has fill*/
                 Linearray[band,line] = 1
                 num_filllines[band] = num_fillines+1
            }
         Else
            Printf("CRC check unsuccessful')
}
Filltotal[band] = num_fillines;
}
```

3. Output linearray and flag each corresponding AM value
4. Output Fill total to DB for trending

4.4.2 Impulse Noise Characterization

4.4.2.1 Background/Introduction

Impulse Noise (IN) is a randomly occurring aperiodic noise that appears as a sample with signal value notably different (far in excess of that expected due to Total Noise) from values exhibited by its nearest neighbors. Sources of IN can include the following:

- 'single-event upsets' (SEUs), i.e., significant increases in the analog output signal produced by a detector due to the effects of charged particles striking the detector array.
- 'bit flips,' i.e., an 'on' bit that is recorded as 'off' or vice versa, due to transmission/recording errors affecting the data stream. These errors are unlikely for Landsat 8 and 9 sensors since the communication with the spacecraft is checked with CRCs and can be retransmitted if an error is detected.

To detect IN artifacts in Level 0 image and calibration data, this algorithm calculates the absolute difference between a pixel value and a median filter value computed using that pixel's nearest neighborhood. This difference is then compared to a threshold value determined from the surrounding pixel values, detector total noise level and minimum detectable IN (IN Limit). Though a linear filter is faster, a median filter is more robust at predicting the true signal value. The median filter is one-dimensional, applied in the image pixel column direction. The affected pixel locations and their values are recorded in the LMASK to allow exclusion from further radiometric processing.

This algorithm is run only on homogenous data sources, i.e., dark shutter bias or lunar data. In addition, the algorithm is run on night image, lamp and solar OLI data and blackbody and deep space TIRS data. The data are processed serially, detector-by-detector, band after band.

4.4.2.2 Inputs

Description	Symbol	Units	Level	Source	Type
Level 0 data: - Dark Earth (OLI only) - Deep space - Shutter - Lamp (OLI only) - Solar (OLI only) - Lunar - Blackbody (TIRS only)	Q	DN	N_{band} x N_{SCA} x N_{det} x N_{frame}		Int
Median Filter Width	MFW	Pixels	N_{band}	CPF	Int
Inoperable detectors			N_{band} x N_{SCA} x N_{det}	CPF	Int
Noise Estimation Outliers	NEO	%	N_{band}	CPF	Float
IN Limit		DN	N_{band}	CPF	Int
Dropped Frames			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int

4.4.2.3 Outputs

Description	Symbol	Units	Level	Target	Type
Number of pixels affected by IN			N_{band} x N_{SCA} x N_{det}	Db	Int
IN Pixel Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
IN Pixel value		DN	N_{band} x N_{SCA} x N_{det} x N_{frame}	Report	Int
IN Neighbor-Pixel values	DNb, DNa	DN	N_{band} x N_{SCA} x N_{det} x N_{frame}	Report	Int

4.4.2.4 Options

A parameter (switch) in work order should provide a possibility to generate a report file, if desired.

4.4.2.5 Procedure

For each band, each detector (d):

1. Ignore pixels corresponding to inoperable detectors, as well as those flagged in LM as dropped frames.
2. Append floor(MFW/2) samples to the beginning and the end of the image, to enable characterization of the pixels at, or close to the beginning and end of the image (pixels for which the filter would fall off the edge of the image).
    a. Copy floor(MFW/2) samples starting from sample #2 of original image to the added samples at the beginning of image.
    b. Copy floor(MFW/2) samples ending in the second last sample of original image to the added samples at the end of image.
3. Compute Median-Filter Value (*Q'*) within the MFW for each pixel (i). The median filter is one-dimensional, applied in the image pixel column direction. For example, for MFW=5,

$$Q'_{d,i} = Median(Q_{d,i-2}, Q_{d,i-1}, Q_{d,i}, Q_{d,i+1}, Q_{d,i+2})$$

a. To characterize pixels surrounding dropped frames, ignore dropped frames. If, for example, sample $Q_{d,i+1}$ were from a dropped frame, compute the Median-Filter value as.

$$Q'_{d,i} = Median(Q_{d,i-2}, Q_{d,i-1}, Q_{d,i}, Q_{d,i+2}, Q_{d,i+3})$$

4. Calculate the difference between the pixel value (Q) and its Q' for each pixel (i)

$$A(i) = Q(i) - Q'$$

5. Sort the calculated difference in ascending order for each detector and, just for the detector noise estimation, remove 2% of points from each side

$$As(d) = sort(A(i))$$
$$Asort(d) = As(0.02*length(As) : 0.98*length(As):length(As))$$

6. Estimate the Total Detector Noise

$$TotalNoise = stdev(Asort)$$

7. Calculate the absolute difference between DN values of the neighboring pixels.

$$Diff(i) = | Q(i+1) - Q(i-1) |$$

8. If Diff(i) is:
   a. greater than twice the Total Detector Noise, then

$$Threshhold = Diff(i) * IN\_Limit(band) / 2$$

   b. less than, or equal to twice the Detector Noise, then

$$Threshold = Total\_Noise(d) * IN\_Limit(SCA)$$

9. Identify an occurrence of IN for each pixel where |A(i)| > Threshold and store in the database the number of pixels affected by IN. If the report file generation is selected as an option in the work order, store into the report file the scene type, band number, SCA number, detector number, frame number (i), and actual output values of the corrupted pixels and their neighbors.
10. Remove from image data the samples added in step 2
11. Update the label mask to include the pixels identified as corrupted by IN.

4.4.3 Saturated Pixel Characterization

4.4.3.1 Background/Introduction

This algorithm flags pixels in data that have saturated digital counts. Two types of saturation can be observed in the OLI data: digital and analog. In TIRS data, at least

one type (digital) is expected, but there might also exist a similarly defined analog saturation level. This algorithm is applicable whether one or both types of saturation should exist.

Digital saturation occurs when the analog input signal represents a radiance level outside of the range that the detector's Analog/Digital (A/D) converter can properly process. The result is that the corresponding digital output is set to 0 DN at the lower end (Digital Low Saturation) and (2^N-1) DN at the upper end (Digital High Saturation) of dynamic range, where N is the signal quantization in bits.

Analog High Saturation occurs when the input signal exceeds the maximum input radiance to which the instrument responds (approximately) linearly or according to any other predefined function. A similar situation occurs at low saturation levels, defining the Analog Low Saturation. The analog saturation levels are determined either from prelaunch measurements or on-orbit characterization and saved in CPF.

The saturated output counts have unknown corresponding radiances and using these data can lead to erroneous research results. Therefore, it is essential to flag saturated pixels for appropriate handling in subsequent processing.

4.4.3.2 Inputs

Description	Symbol	Units	Level	Source	Type
All Level 0 data	Q	DN	N_{band} x N_{SCA} x N_{det} x N_{frame}		Int
Impulse Noise Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
Dropped Frame Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
Significant Bit Flag (least, 'L', or most, 'M', significant 12 bits)			1	metadata	Char
Analog Low Saturation Level		DN	N_{band} x N_{SCA} x N_{det}	CPF	Int
Analog High Saturation Level		DN	N_{band} x N_{SCA} x N_{det}	CPF	Int
Digital Low Saturation Level		DN	N_{band} x N_{SCA} x N_{det}	CPF	Int
Digital High Saturation Level		DN	N_{band} x N_{SCA} x N_{det}	CPF	Int

4.4.3.3 Outputs

Description	Symbol	Units	Level	Source	Type
Digital Low Saturation Pixel Total			N_{band} x N_{SCA} x N_{det}	Db	Int
Digital High Saturation Pixel Total			N_{band} x N_{SCA} x N_{det}	Db	Int
Digital Low Saturation Pixel Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
Digital High Saturation Pixel Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
Analog Low Saturation Pixel Total			N_{band} x N_{SCA} x N_{det}	Db	Int

Description	Symbol	Units	Level	Source	Type
Analog High Saturation Pixel Total			N_{band} x N_{SCA} x N_{det}	Db	Int
Analog Low Saturation Pixel Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM & Report	Int
Analog High Saturation Pixel Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM & Report	Int
Analog Low Saturation Pixel Values		DN	N_{band} x N_{SCA} x N_{det} x N_{frame}	Report	Int
Analog High Saturation Pixel Values		DN	N_{band} x N_{SCA} x N_{det} x N_{frame}	Report	Int

4.4.3.4 Options

A parameter (switch) in work order provides the option to generate a report file.

4.4.3.5 Procedure

For each detector, d, including inoperable detectors, for all bands and SCAs:

1. Ignore pixels flagged in LM as dropped frames or corrupted with impulse noise. Thus, if a pixel identified as impulse noise affected has a value equal to saturation value, it will not be identified as saturated.
2. Using analog saturation levels from CPF, find and flag each saturated pixel. Analog saturated pixels are those pixels that have DN values below Analog Low Saturation Level, but higher than the Digital Low Saturation Level, and pixels that have DN values above Analog High Saturation Level, but lower than the Digital High Saturation Level. Thus, if analog and digital saturation levels are set to the same value, no analog saturation will be detected. Use different flags for low and high saturation.
3. Using digital saturation levels, 0 DN for low and (2^N-1) DN for high saturation, find and flag all digitally saturated pixels. Use different flags for low and high saturation and different flags than used to identify analog saturation levels.
4. Output all flags to corresponding pixel locations in LM file
5. Record the number of low and high saturated pixels per detector to the database for both digital and analog types of saturation. Record pixel values from analog saturated pixels to the report file, if the report file is generated.
6. Save these scene summary statistics to the database

4.4.4 Histogram Statistics Characterization

4.4.4.1 Background/Introduction

This algorithm serves as a general purpose algorithm occurring at different locations in the L1R processing flow. It supports characterization of all active detectors, including the inoperable ones, by computing statistics from single images or collects up to ~4.5 minutes long: minimum, maximum, mean, standard deviation, skewness, and kurtosis. In addition, the algorithm calculates for each detector the mean of squared pixel values and the adjacent detector correlations. For Earth scenes, the means and the standard deviations per SCA and per band are computed. To calculate these values, the input scene data need to be nominally spatially aligned. All results are stored in the database

and used in other algorithms. For the OLI panchromatic band, all statistics are calculated and saved separately for odd and even frames.

4.4.4.2 Inputs

Description	Symbol	Units	Level	Source	Type
Scene data at any processing level, including blind detectors and video reference pixels (VRP)					
Q			N_{band} x N_{SCA} x N_{det} x N_{frame}	Int or Float	
Impulse Noise Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
Dropped Frame Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
Saturated Pixel Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int

4.4.4.3 Outputs

Description	Symbol	Units	Level	Target	Type
Detector Minimum response	Q_{min}	DN	N_{band} x N_{SCA} x N_{det}	Db	Float
Detector Maximum response	Q_{max}	DN	N_{band} x N_{SCA} x N_{det}	Db	Float
Detector Mean	$\overline{Q}$	DN	N_{band} x N_{SCA} x N_{det}	Db	Float
Detector Standard deviation	σ	DN	N_{band} x N_{SCA} x N_{det}	Db	Float
Detector Skewness	γ_1		N_{band} x N_{SCA} x N_{det}	Db	Float
Detector Kurtosis	γ_2		N_{band} x N_{SCA} x N_{det}	Db	Float
Number of valid frames for a detector	N_{valid_pixels}	Pixels	N_{band} x N_{SCA} x N_{det}	Db	Int
Detector Mean squared response	$\overline{Q^2}$	DN^2	N_{band} x N_{SCA} x N_{det}	Db	Float
Adjacent detector correlation	ρ		N_{band} x N_{SCA} x N_{det}	Db	Float
SCA Mean (for Earth scenes only)	$\overline{Q}_{SCA}$	DN	N_{band} x N_{SCA}	Db	Float
SCA Standard deviation (for Earth scenes only)	σ_{SCA}	DN	N_{band} x N_{SCA}	Db	Float
SCA average number of frames (for Earth scenes only)	N_{SCA_frames}		N_{band} x N_{SCA}	Db	Float
Band Mean (for Earth scenes only)	$\overline{Q}_{band}$	DN	N_{band}	Db	Float
Band Standard deviation (for Earth scenes only)	σ_{band}	DN	N_{band}	Db	Float
Band average number of frames (for Earth scenes only)	N_{band_frames}		N_{band}	Db	Float
Position in processing flow (RPS level)			1	Db	String

Description	Symbol	Units	Level	Target	Type
Linearization LUT version			1	Db	String

Note: For the OLI panchromatic band, all output values need to be generated and saved separately for odd and even minor frames.

4.4.4.4 Options

Typically, these data will be stored in the characterization database. For stand-alone processing, the individual detector statistics may be output to a summary report with a header containing start date and time of acquisition, end date and time of acquisition, processing date and time, calculated frame rate, filename and entity ID. A report generation should be selectable in work order.

4.4.4.5 Procedure

For each active detector, *d*, including inoperable detectors, blind detectors and VRPs, for all bands and SCAs, except for the OLI panchromatic band:

1. Obtain number of valid pixels, $N_{valid\_pixels}\ (d)$, in image or collect. Pixels identified in the label mask as saturated, impulse noise affected, or parts of dropped frames, as well as filled pixels used to generate detector offsets and pixels corresponding to the Number of frames to be skipped at the top and bottom of the image, are considered invalid and need to be taken out of calculations. The symbol *l* is used to denote the list of valid pixels.

2. Find the minimum of valid pixel values, *l*:

$$Q_{\min}(d) = \min\left(Q(l,d)\right)$$

3. Find the maximum of the valid pixel values, *l*:

$$Q_{\max}(d) = \max\left(Q(l,d)\right)$$

4. Calculate mean as:

$$\overline{Q}(d) = \frac{1}{N_{valid\_pixels}(d)} \sum_{l=1}^{N_{valid\_pixels}} Q(l,d)$$

5. Calculate standard deviation as:

$$\sigma(d) = \sqrt{\frac{1}{N_{valid\_pixels}(d)} \sum_{l=1}^{N_{valid\_pixels}} (Q(l,d) - \overline{Q}(d))^2}$$

6. Calculate skewness as:

$$\gamma_1(d) = \frac{\sum_{l=1}^{N_{valid\_pixels}} (Q(l,d) - \overline{Q}(d))^3}{N_{valid\_pixels}(d) \times \sigma(d)^3}$$

If $\sigma(d) = 0$, set $\gamma_1 = 0$.

7. Calculate kurtosis as:

$$\gamma_2(d) = \frac{\sum_{l=1}^{N_{valid\_pixels}} (Q(l,d) - \overline{Q}(d))^4}{N_{valid\_pixels}(d) \times \sigma(d)^4} - 3$$

If $\sigma(d) = 0$, set $\gamma_2 = 99999$.

8. Calculate correlation between each detector (image column) and the neighbor on its right side using only pixel pairs where adjacent pixels from both detectors are valid. This step needs to be performed for all detectors except the last one on each SCA.

$$\rho_{d,d+1}(d) = \frac{1}{N_{valid\_both}} \sum_{l=1}^{N_{valid\_both}} Q(l,d)Q(l,d+1)$$

where N_{valid_both} is the number of valid adjacent pixel pairs. A pixel pair is valid if both adjacent pixels in spatially aligned image, generated by two neighboring detectors, are valid.

9. Calculate mean squared response:

$$\overline{Q^2}(d) = \frac{1}{N_{valid\_both}} \sum_{l=1}^{N_{valid\_both}} Q(l,d)^2$$

10. Save results to the database. Generate a report file, if that option selected in work order.

For each active detector, *d*, in the OLI panchromatic band:

Repeat the steps 1 to 10, but separately for odd and even frames.

If the processed scene is an Earth scene, for each image band (OLI bands 1 to 9 and TIRS bands 10 and 11):

1. Calculate the SCA mean as:

$$\overline{Q}_{SCA} = \frac{\sum_{d=1}^{N_{\det ectors}} N_{valid\_pixels}(d) \times \overline{Q}(d)}{\sum_{d=1}^{N_{\det ectors}} N_{valid\_pixels}(d)}$$

2. Calculate the SCA standard deviation as:

$$\sigma_{SCA} = \sqrt{\frac{\sum_{d=1}^{N_{\det ectors}} N_{valid\_pixels}(d)\left(\sigma(d)^2 + \overline{Q}(d)^2\right) - \left(\sum_{d=1}^{N_{\det ectors}} N_{valid\_pixels}(d)\right) \times \overline{Q}_{SCA}{}^2}{\sum_{d=1}^{N_{\det ectors}} N_{valid\_pixels}(d)}}$$

3. Calculate the average number of frames for SCA as:

$$N_{SCA\_frames} = \frac{\sum_{d=1}^{N_{\det ectors}} N_{valid\_pixels}(d)}{N_{\det ectors}}$$

4. Calculate the band mean as:

$$\overline{Q}_{band} = \frac{\sum_{s=1}^{N_{SCAs}} N_{SCA\_frames}(s) \times \overline{Q}_{SCA}(s)}{\sum_{s=1}^{N_{SCAs}} N_{SCA\_frames}(s)}$$

5. Calculate the band standard deviation as:

$$\sigma_{band} = \sqrt{\frac{\sum_{s=1}^{N_{SCAs}} N_{SCA\_frames}(s)\left(\sigma_{SCA}(s)^2 + \overline{Q}_{SCA}(s)^2\right) - \left(\sum_{s=1}^{N_{SCAs}} N_{SCA\_frames}(s)\right) \times \overline{Q}_{band}{}^2}{\sum_{s=1}^{N_{SCAs}} N_{SCA\_frames}(s)}}$$

6. Calculate the average number of frames for band as:

$$N_{band\_frames} = \frac{\sum_{s=1}^{N_{SCAs}} N_{SCA\_frames}(s)}{N_{SCAs}}$$

7. Save these scene summary statistics to the database

4.4.5 Temperature Sensitivity Characterization

4.4.5.1 Background/Introduction

Focal plane temperature variations affecting gain and offsets, were considered for correction during Landsat and EO-1 ALI processing. A similar correction for gain and bias will be available for both OLI, and TIRS during LPGS data processing.

The acquisitions using the on-board cal sources, and their associated focal plane thermistor data, are used to determine and characterize the temperature sensitivity coefficients (C_T) used in correction. For OLI, the (primary) diffuser and (primary) stim lamp collects, and average of 2 focal plane thermistor readings will be used. For TIRS, the OBC collects and the average of 4 focal plane thermistor readings will be used. The regression slopes of response to temp will provide the coefficients.

Note, while the Focal Plane temperatures are identified, it is conceivable that Focal Plane Electronics temps could also be used.

Note, while all thermistors are assumed to operate nominally at launch, postlaunch these thermistors will be continuously characterized i.e., temperatures will be trended and evaluated per interval and over the mission life (“globally”). Should this characterization reveal outliers and/or any thermistors falling “out of family,” that thermistor will be evaluated for exclusion. It is assumed the associated thermistor readings have been converted to degs C and any time offset in the temperature telemetry has been corrected prior to extraction into this tool.

4.4.5.2 Inputs

Description	Symbol	Units	Level	Source	Type
Focal Plane Temperatures (OLI)	FPM_7_TEMP_CELSIUS (FPM 7 Temp)= T_1 FPM_14_TEMP_CELSIUS (FPM 14 Interface Temp) = T_2	C		Db	Float
Focal Plane Temperatures (TIRS)	FP_F2_FINE_SENSOR_1_CELSIUS =T1 FP_F4_FINE_SENSOR_3_CELSIUS =T2 FP_F6_FINE_SENSOR_1_CELSIUS =T3 FP_F7_FINE_SENSOR_2_CELSIUS =T4	C		Db	Float
OLI Diffuser or Lamp (PRIMARY) Gains	$G_{OLI\text{-}S}$ or $G_{OLI\text{-}L}$	DN/ W/m^2 -mu-sr	N_{band} x N_{det}	Db	Float
TIRS OBC Gains	G_{TIRS}	DN/ W/m^2 -mu-sr	N_{band} x N_{det}	Db	Float
Start Time, and Stop Times	T_s, T_f	Secs			

4.4.5.3 Outputs

Description	Symbol	Units	Level	Source	Type
OLI Temp (Means and Stdevs)	$<T_n>$, σT_{n-} where n=1,2	C			Float
OLI Mean Temp	<T>	C			Float
OLI Temp Sensitivity Coeff and uncertainty	C_{T-H} ΔC_{T-H}	DN/C	N_{band} x N_{SCA} x N_{det}	CPF	Float
TIRS Temp (Means and Stdevs)	$<T_n>$ $\sigma T_{n-Stdev}$ where n=1,2,3,4	C			Float
TIRS Mean Temp	<T>	C			Float
TIRS Temp Sensitivity Coeff and uncertainty	C_{T-O} ΔC_{T-O}	DN/C	N_{band} x N_{SCA} x N_{det}	CPF	Float

4.4.5.4 Options

OLI: Lamp or Diffuser Data Input; Time Range of Data,
TIRS: OBC temp; Time Range of Data

4.4.5.5 Procedure

Preparation: In advance of running this algorithm the variation in the various focal plane related temperatures should be trended. If variation is observed, then similar variations in these temperatures should be looked for in calibration acquisitions (lamp and solar for OLI and OBC at a fixed BB temp for TIRS). If variation is observed in the calibration acquisitions, the appropriate temperatures and instrument responses should be extracted from the IAS database.

For each calibration interval selected

For each thermistor (T_n)

1. Generate temperature avg's ($<T_n>$) and stdev's, (σT_n)

2. Reject any "outlier" thermistor and set thermistor flags to 0 in CPF.

End; thermistor loop

3. Average across all valid thermistors for the interval (<T>)

End; cal interval

4. For a selected instrument calibration type, over a give range of start (T_s) and stop times (T_f) , extract and regress the instrument gains ($G_{OLI-S/L}$ or G_{TIRS}) to the avg focal plane temperature (<T>)

5. Calculate the temperature sensitivity coefficient C_T (where C_T is inverse of correction factor per Jackson and Robinson, (1985),) by ratioing the slope of the regression to the intercept, when the regression is significant. A value of '0' translates to no temperature correction.

4.4.6 Temperature Sensitivity Correction

4.4.6.1 Background/Introduction

Focal plane temperature variations affecting gain and offsets, were considered for correction during Landsat and EO-1 ALI processing. A similar correction for gain and bias will be available for both OLI, and TIRS during LPGS data processing. There are 2 thermistors on the OLI focal plane. The TIRS focal plane has 8 thermistors, only 4 of which will be provided as part of the ancillary data.

This algorithm uses an average of each set of focal plane temperatures to derive the temperature correction. This is a linear correction and is done on a per-detector basis. These thermistors are available in the ancillary data and are updated every 1 second for OLI. (This is anticipated to be the same for TIRS). This algorithm assumes that these thermistors readouts have been converted to K and any time offset in the temperature telemetry has been corrected prior to usage in this algorithm.

4.4.6.2 Inputs

Description	Symbol	Units	Level	Source	Type
Temp Sensitivity Coefficient	C_T	-	N_{band} x N_{SCA} x N_{det}	CPF	Float
Scene Focal Plane Temperatures (OLI)	FPM_7_TEMP_CELSIUS (FPM 7 Temp)= T_1 FPM_14_TEMP_CELSIUS (FPM 14 Interface Temp) = T_2	C	Per Interval	Ancillary	Float
Scene Focal Plane Temperatures (TIRS)	FP_F2_FINE_SENSOR_1_CELSIUS (SCA-A Edge Sensor)=T1 FP_F4_FINE_SENSOR_3_CELSIUS (SCA-B Edge Sensor)=T2 FP_F6_FINE_SENSOR_1_CELSIUS (Center FPA Sensor) =T3 FP_F7_FINE_SENSOR_2_CELSIUS (Center FPA Edge Sensor) =T4	C	Per Interval	Ancillary	Float
Reference Temperature	T_{Ref}	K	Per Instrument	CPF	Float
Thermistor_Flag	$T_{Flag\_}$	-	Per Instrument Thermistor i.e., 2 OLI, 4 TIRS	CPF	Int

4.4.6.3 Output

Description	Symbol	Units	Level	Target	Type
Correction Factor	CF_T	-	N_{band} x N_{SCA} x N_{det}	Gain Application	Float

4.4.6.4 Options

The invocation of this algorithm and the application of CF_T, can be performed at any stage prior applying gain during LPGS processing.
By default, this correction will be OFF during LPGS processing as specified via work order parameter.

4.4.6.5 Procedure

1) For each set of instrument thermistors #1 to #n, extract "scene-equivalent" temperatures based upon n-thermistor sample times and scene-start/stop times, where n=2 for OLI and n=4 for TIRS

$T_1 = T_1$ where $((t_1 \text{ GE } ts_i) \text{ and } (t_1 \text{ LE } ts_f))$
…
$T_n = T_n$ where $((t_n \text{ GE } ts_i) \text{ and } (t_n \text{ LE } ts_f))$

Where ts_i = Scene Time initial
ts_f = Scene Time final
t_1 = Thermistor-#1 times
t_n = Thermistor-#n times

2) Average each Focal Plane temperatures (from Step 1) and derive single average Focal Plane temperature (T_{FP}) i.e.

$T_{1\,AVG}$ = Sum of all T_1 / Total # of T_1
…
$T_{n\,AVG}$ = Sum of all T_n / Total # of T_n

3) Multiply each single avg Focal Plane temp by its corresponding Thermistor Flag value i.e.,

$T_{1AVG} = T_{1AVG} \times T_{1Flag}$
…
$T_{nAVG} = T_{nAVG} \times T_{nFlag}$

Where

T_{1Flag} = 1 (default)
…
T_{nFlag} = 1 (default)

Note, the value of each Thermistor flag is "good" (=1). Should a thermistor become "bad ," its corresponding Thermistor flag will be set = 0.

4) Average all “good” thermistor avg values from Step 3) i.e.,

$T_{FP} = (T_{1AVG} + \ldots + T_{nAVG})/n$

5) For each Band, SCA and detector, using the equation in Step 4) and other input parameters, to derive and output the Correction Factor (CF_T) i.e.,

$CF_T = [(1 + C_T*T_{FP})/(1 + C_T*T_{Ref})]$

Where C_T= Temperature sensitivity coefficient
(This is the inverse coef used by Jackson et al., 1985)
T_{FP} = Focal Plane temperature (K)
T_{Ref} = Reference temperature (K)

Suggested implementation during PGS processing
1) Run algorithm prior to “Apply Gain” algorithm
2) Input TCorr to “Apply Gain” algorithm and multiply by the applied gain.

4.4.7 Gain Application

4.4.7.1 Background/Introduction

Generation of Level 1 Radiometric (Corrected) products result in an estimated in-band radiance product in W/m^2-sr-µm. This conversion occurs in 3 steps/algorithms; bias subtraction, non-linearity correction and gain (Absolute, SCA Relative, and Detector Relative) application. Processing options allow the generation of intermediate calibrated products – e.g., with bias correction only, or non-linearity correction only, etc.

The Gain Application algorithm addresses the final step in generating the L1r radiance product. The type of gain application can be selected by parameters within the processing work order. Absolute gain application applies the same gain to all detectors within a band and SCA. SCA Relative gain applies a different gain to each SCA with all detectors within an SCA having the same gain, and Detector Relative gain application involves a different gain applied to each detector.

Another possibility in gain application is the optional processing of gain temperature sensitivity corrections. The Temp Sensitivity Correction Algorithm description describes the calculation of the sensitivity correction parameters.

4.4.7.2 Input

Description	Symbol	Units	Level	Source	Type
Scene (bias corrected, linearized)	Q	DN	N_{band} x N_{SCA} x N_{det} x N_{frame}	Response Linearization	Float
Reference Absolute Gain	G_{ref}	DN/(w/m^2 sr um)	N_{band}	CPF	Float
Absolute Gain	G_{abs}	DN/(w/m^2 sr um)	N_{band}	CPF	Float

SCA Relative Gain	G_{SCArel}	Unitless	N_{band} x N_{SCA}	CPF	Float
Detector Relative Gain	G_{Detrel}	Unitless	N_{band} x N_{SCA} x N_{det}	CPF	Float
Temperature Sensitivity Coefficients	CF_T	Unitless	N_{band} x N_{SCA} x N_{det}	Temperature Sensitivity Correction	Float
Absolute Gain Option			1	Work Order	Boolean
Apply SCA Relative Gain Flag			1	Work Order	Boolean
Apply Detector Relative Gain Flag			1	Work Order	Boolean
Apply Temperature Sensitivity Flag			1	Work Order	Boolean

4.4.7.3 Output

Description	Symbol	Units	Level	Source	Type
Scene (bias corrected, linearized)	Q	DN	N_{band} x N_{SCA} x N_{det} x N_{frame}	Response Linearization	Float

4.4.7.4 Options

- Apply Absolute Gain Option
  - No absolute gains (default off)
  - Apply absolute gains (default on)
  - Apply reference absolute gains (default off)
- Apply SCA relative gain (default on)
- Apply detector relative gain (default on)
- Apply temperature sensitivity correction (default off)

4.4.7.5 Procedure

The first step in gain application is the decision of which gains to apply. Work order parameters should allow the operator to separately choose whether or not to apply absolute, SCA relative, and/or detector relative gains and/or temperature sensitivity correction. Once those parameters are parsed, the respective gain parameters should be obtained from the CPF.

If no gain application is desired skip to step 3, else
for each band

1. Combine separate gains.

$$G_{com}(d) = G_{abs}(B) * G_{SCArel}(S) * G_{Detrel}(d) / CF_T(d) \quad (1)$$

where $G_{com}(d)$ = Combined Gain for detector d,
$G_{abs}(B)$ = Absolute Gain for Band B,
$G_{SCArel}(SCA)$ = SCA Relative Gain for SCA S,
$G_{Detrel}(d)$ = Detector Relative Gain for detector d,
and $CF_T(d)$ = Temperature Sensitivity Coefficient for detector d.
if a certain gain is not needed, a value of "1" can be substituted.

2. Apply the combined gain to the bias corrected and response linearized scene.

$$L(d) = Q(d) / G_{com}(d) \quad (2)$$

where $L(d)$ = Output radiance value for detector d,
$Q(d)$ = Input DN value (linearized and bias corrected) for detector d,

3. Set all Apply flags accordingly.

4.4.8 L1R SCA Stitching

4.4.8.1 Background/Introduction

During product generation the OLI and TIRS data are radiometrically processed on an SCA by SCA basis, but stitching of data from all SCAs to the instrument's full-field-of-view product for each band does not occur until Geometric Processing. Certain radiometric assessments at various stages in the processing flow, e.g., Non-uniformity Characterization, need to be performed on the entire band images.

This algorithm uses the SCA offsets to nominally spatially align all SCAs within each band and stitches the SCAs together to generate L1R single band images. The generated single band images can then be combined to form nominally aligned multispectral images. The algorithm assumes that the input data within each SCA are nominally spatially aligned, i.e., the odd/even detector offsets and the offsets for the selected non-primary detectors have been applied. Further, it is assumed that information about the number of imaging detectors per SCA (SCA width, in pixels) and the SCA length, in pixels, are available.

The stitching can be accomplished by handling the SCA overlap regions in several ways:

- Method 1 – No Overlap; stitching without overlapping the SCAs (Figure 4-83)
- Method 2 – Left Overlap; SCA1 is complete and all other SCAs miss the overlap region on their left side in the image (Figure 4-84)
- Method 3 – Right Overlap; the last SCA is complete and all other SCAs miss the overlap region on their right side in the image (Figure 4-85)
- Method 4 – Half-Half, SCA1 misses half of the overlap region on its right side, the last SCA misses half of the overlap region on its left side, and all other SCAs miss half of the overlap region on their both sides (Figure 4-86).

4.4.8.2 Inputs

Description	Symbol	Units	Level	Source	Type
Scene L1R data, except the blind band	L1Rp	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA} x N_{det} x N_{frame}		Float
SCA overlap width	SCA_Overlap_Width	Pixels	N_{band}	CPF	Int

Description	Symbol	Units	Level	Source	Type
Stitching method	Method		1	Work Order	Int or String
Detector offsets	O_{det}	Pixels	N_{band} x N_{SCA} x N_{det}	CPF	Float
Along-track Legendre Polynomial Coefficients			N_{band} x N_{SCA} x 4	CPF	Float
Nominal Orbit Radius	r_o	km	1	CPF	Float
Orbital Period	T_o	s	1	CPF	Float
Semi Major Axis	r_{eq}	m	1	CPF	Float
Semi Minor Axis	r_p	m	1	CPF	Float
Eccentricity	E		1	CPF	Float
Earth Angular Velocity	V_g	Radians/s	1	CPF	Float
Nominal Frame Time	t_s	ms	N_{sensor}	CPF	Float
Scene Center Latitude	ϕ	Degrees	1	Scene Metadata	Float
Nominal Scene Elevation*	d	km	1	DEM	Float
SCA Offsets	O_{SCA}	Pixels	N_{band} x N_{SCA}	CPF	Int

*The nominal scene elevation can be the average of the minimum and maximum elevation of the scene.

4.4.8.3 Outputs

Description	Symbol	Units	Level	Target	Type
Nominally aligned L1R band image (L1Rp)	L1R	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA} x N_{det} x N_{frame}		Float
Stitching method			1	metadata	Int or String

4.4.8.4 Options

The stitching method and list of bands to be stitched are selected through the work order. Additionally, the option to use just the SCA offsets from the CPF (O_{SCA}), without doing the extra calculations, enables non-Earth scenes to be aligned manually.

4.4.8.5 Procedure

1. For Earth scenes, calculate the SCA_Offset according to the algorithm described below. For non-Earth scenes, the O_{SCA} straight from the CPF is used.
2. Calculate the stitched product length as:

   L1Rp_Length = SCA_Length + max(SCA_Offset)

   The SCA length represents the worst case scenario; it is the length of an SCA after the maximum possible detector offsets are applied (SCA_Length = N_{frames} + max(round(O_{det}))).

3. For the sensor specific number of SCAs per band, N_{SCA}, calculate width of the stitched product:
   a. If the Stitching method is “No Overlap”

L1Rp_Width = N_{SCA} x SCA_Width

b. otherwise

L1Rp_Width = N_{SCA} x SCA_Width – (N_{SCA}-1) x SCA_Overlap_Width

Note that this equation assumes that all SCA overlaps have the same width. If that assumption proves inadequate, each SCA to SCA overlap width will need to be accounted for individually.

4. For each band:

   a. Allocate an array, L1Rp, of size L1Rp_Length x L1Rp_Width and fill it with zeros.

   b. Load all SCAs

   c. If the Stitching method is “No Overlap” (see Figure 4-83)
      i. For n = 1 to N_{SCA},
         1. Copy the SCA(n) data to the sub-array of the L1Rp array defined in the 1-based coordinate system as:

	x (Column)	**y (Row)**
Upper-Left Corner	(n-1) x SCA_Width + 1	SCA_Offset(n) + 1
Lower-Right Corner	n x SCA_Width	SCA_Offset(n) + SCA_Length

Table 4-52. No Overlap SCA Stitching Positions

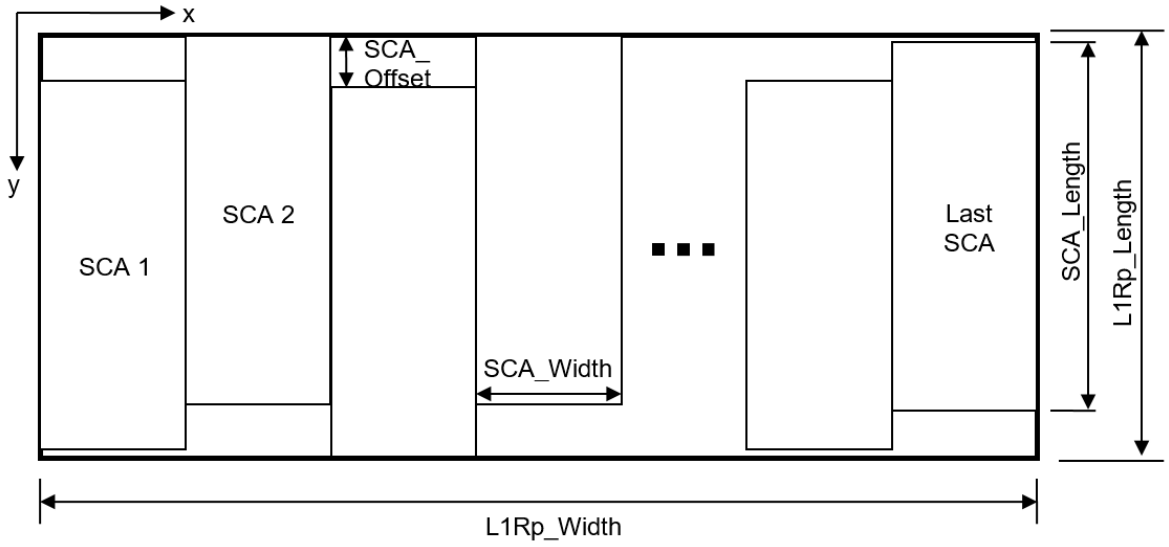

Figure 4-83. No Overlap SCA Stitching

d. If the Stitching method is “Left Overlap” (see Figure 4-84)
    i. For n = N_{SCA} to 1 with step of -1,
        1. Copy the SCA(n) data into the sub-array of the L1Rp array defined in the 1-based coordinate system as:

	x (Column)	y (Row)
Upper-Left Corner	L1Rp_Width – (N_{SCA} – n) x (SCA_Width – SCA_Overlap_Width) - SCA_Width + 1	SCA_Offset(n) + 1
Lower-Right Corner	L1Rp_Width – (N_{SCA} – n) x (SCA_Width – SCA_Overlap_Width)	SCA_Offset(n) + SCA_Length

Table 4-53. Left Overlap SCA Stitching Positions

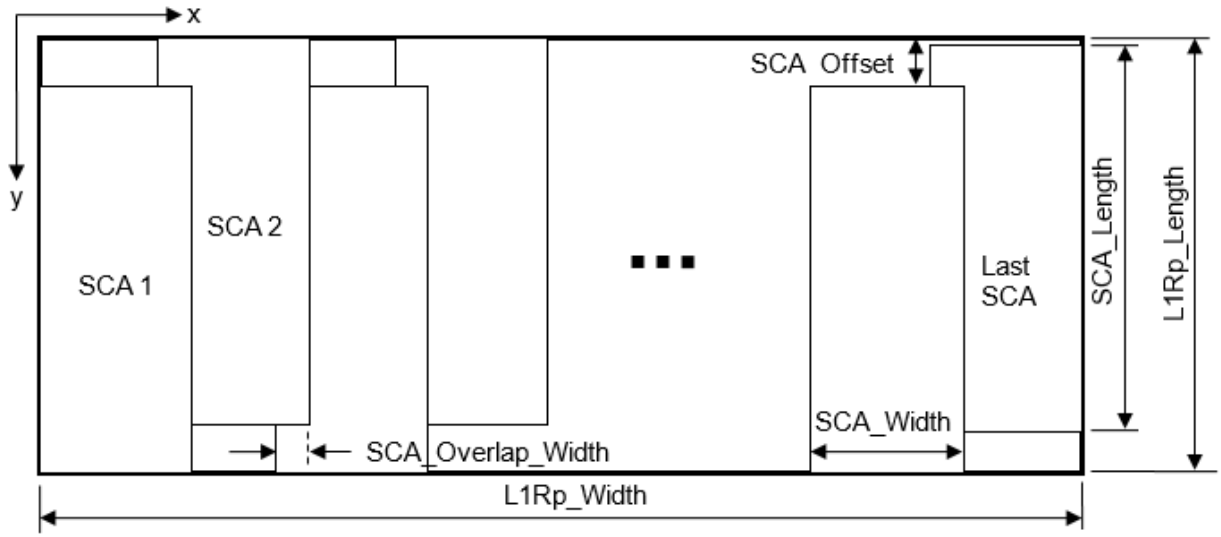

Figure 4-84. Left Overlap SCA Stitching

e. If the Stitching method is “Right Overlap” (see Figure 4-85)
    i. For n = 1 to N_{SCA},
        1. Copy the SCA(n) data into the sub-array of the L1Rp array defined (in the 1-based coordinate system) as:

	x (Column)	y (Row)
Upper-Left Corner	(n-1) x (SCA_Width - SCA_Overlap_Width) + 1	SCA_Offset(n) + 1
Lower-Right Corner	n x (SCA_Width - SCA_Overlap_Width)	SCA_Offset(n) + SCA_Length

Table 4-54. Right Overlap SCA Stitching Positions

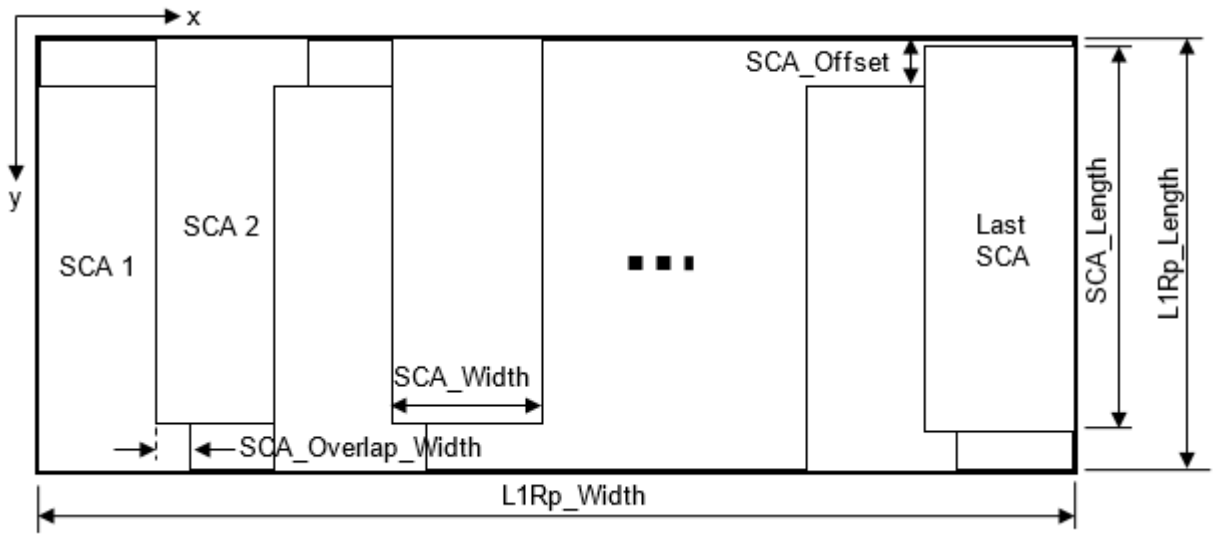

Figure 4-85. Right Overlap SCA Stitching

f. If the Stitching method is “Half-Half” (see Figure 4-86)
   i. Copy the SCA1 data into the sub-array of the L1Rp array defined in the 1-based coordinate system as:

	x (Column)	**y (Row)**
Upper-Left Corner	1	SCA_Offset(1) + 1
Lower-Right Corner	SCA_Width	SCA_Offset(1) + SCA_Length

Table 4-55. Half-Half Overlap SCA1 Stitching Position

   ii. For n = 2 to N_{SCA},
      1. If SCA_Overlap_Width is an odd number
         a. Skip the first (SCA_Overlap_Width + 1) / 2 columns and copy the rest of the SCA(n) data into the sub-array of the L1Rp array defined in the 1-based coordinate system as:

	x (Column)	**y (Row)**
Upper-Left Corner	(n-1) x (SCA_Width - SCA_Overlap_Width) + 1 + (SCA_Overlap_Width + 1) / 2	SCA_Offset(n) + 1
Lower-Right Corner	(n-1) x (SCA_Width - SCA_Overlap_Width) + SCA_Width	SCA_Offset(n) + SCA_Length

Table 4-56. Half-Half Overlap Odd SCA Stitching Position

2. Else
    a. Skip the first SCA_Overlap_Width / 2 columns and copy the rest of the SCA(n) data into the sub-array of the L1Rp array defined in the 1-based coordinate system as:

	x (Column)	y (Row)
Upper-Left Corner	(n-1) x (SCA_Width - SCA_Overlap_Width) + 1 + SCA_Overlap_Width/2	SCA_Offset(n) + 1
Lower-Right Corner	(n-1) x (SCA_Width - SCA_Overlap_Width) + SCA_Width	SCA_Offset(n) + SCA_Length

Table 4-57. Half-Half Overlap Even SCA Stitching Position

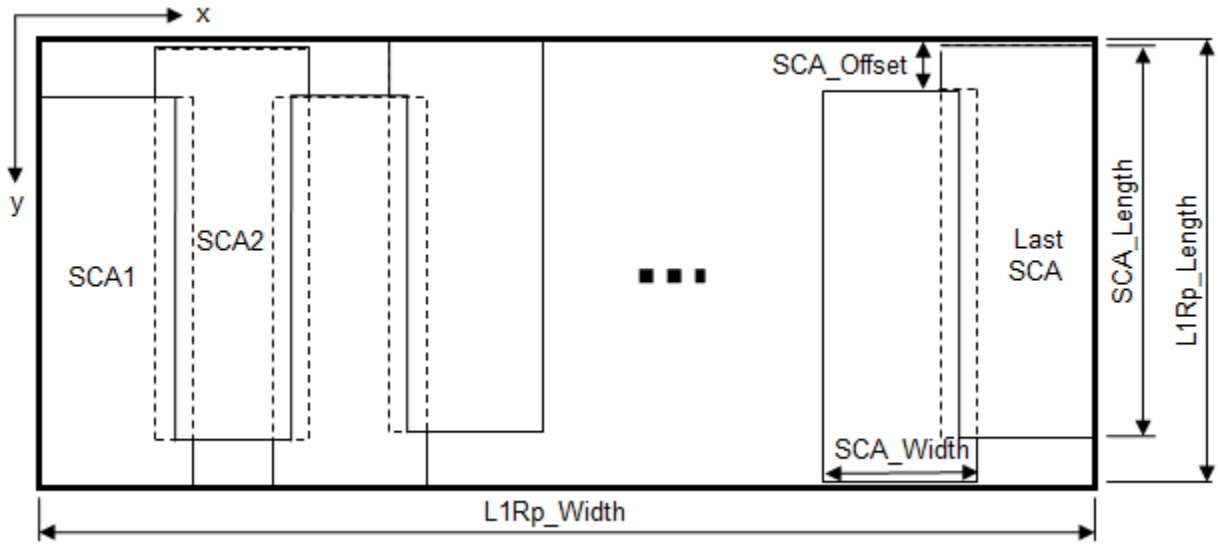

Figure 4-86. Half-Half Overlap SCA Stitching

5. Record the used Stitching method to metadata

4.4.9 Striping Characterization

4.4.9.1 Background/Introduction

Evaluation of the effectiveness of relative gain correction to remove striping is typically performed in a qualitative sense, through visual inspection of imagery before and after correction. This method has several limitations, the primary one being that it relies on subjective human interpretation for the evaluation. In addition, inspection of large numbers of corrected images is not realistic. Consequently, a quantitative characterization of striping is needed.

Algorithms have been developed to quantitatively characterize striping through frequency-domain analyses (mostly FFT-based). These can produce an average

estimate (across a focal plane module) of the amount of striping at the Nyquist frequency (corresponding to detector-to-detector variation). The disadvantage of this approach is that the results provide no real information about striping at a detector level (i.e., which detectors are more sensitive to striping).

This algorithm determines a quantitative metric for the amount of striping present in an image through calculation of spatial-domain statistics for each detector. These statistics are further processed to obtain a "final" striping metric. In the initial development work it has been found that larger values for this metric tend to positively correlate with more visually apparent striping.

4.4.9.2 Inputs

Description	Symbol	Units	Level	Source	Type
Scene	*Q*	DN or W/m^2 sr □m	N_{band} x N_{SCA} x N_{det} x N_{frame}		Float
Saturated pixels			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
Impulse noise			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
Dropped Frames			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
Inoperable detectors			N_{band} x N_{SCA} x N_{det}	CPF	Int
Absolute Gain	G_{abs}		N_{band}	CPF	Float
Striping metric cutoffs			N_{band}	CPF	Float

The Striping Metric Cutoffs have a default of 2% of the standard deviation of "all" the images in the archive. This will be fairly constant, so there is not a need to query the database and calculate it every time. A value in the CPF should function adequately. Initialization of this parameter may be done after 100 images are in the archive/database.

4.4.9.3 Outputs

Description	Symbol	Units	Level	Destination	Type
Overall Striping Metric		DN or W/m^2 sr μm	N_{band}	Db	Float
Detector Striping Metric (optional)			N_{band} x N_{SCA}	Report	Float
Scene Striping Metric (optional)			N_{band} x N_{SCA}	Report	Float

The Overall Striping Metric is a single number measure of the amount of striping found in the image.

The Detector Striping Metric is an N_{det-2} array measure of the amount of the striping in each individual detector.

The Scene Striping Metric is an N_{det-2} x $N_{frames-fill-2}$ measure of the amount of striping at each individual pixel.

4.4.9.4 Options

Write the Overall Striping Metric to database (On by Default)
Summary Report (Off by Default)

a) Detector striping metric
b) Scene striping metric
c) Overall striping metric

4.4.9.5 Procedure

1. Read in the processing parameters.
2. Read in an SCA.
3. Find the difference between every pixel in the image and the average of its two neighbors (left and right). When a pixel in the artifact mask or a sample from an inoperable detector is encountered, the Cross-Track Difference (CTDiff should be set to zero. So a pixel in the artifact mask or a sample from an inoperable detector will cause three entries to be zeros in CTDiff (the pixel itself and its two neighbors).

$$\mathrm{CTDiff} = x_{m,n} - \frac{x_{m,n-1} + x_{m,n+1}}{2}$$

   Where *x* denotes a pixel, and *m* and *n* denote row (frame) and column (detector), respectively. This difference is calculated for all pixels in the image except border pixels (one pixel on all sides).
4. Since scene content will cause the largest magnitudes in CTDiff, we will calculate a homogeneity filter.
   a. The first step is to check the Cross-Track Homogeneity (CTHom) by calculating the difference between pixels in the image on either side of the current pixel.

$$\mathrm{CTHom(m,n)} = \left| x_{m,n-1} - x_{m,n+1} \right|$$

      This difference is calculated for all pixels in the image except border pixels (one pixel on all sides). Whenever a pixel in the artifact mask or a sample from an inoperable detector is encountered, CTHom should be zero, so all pixels in the artifact mask and their left and right adjacent pixels will have zero values in CTHom. Thus, inoperable detectors will cause three columns of zeros in CTHom.
   b. Next we will check the Along-Track Homogeneity (ATHom) by taking the vertical difference between the current pixel and its top and bottom neighbors.

$$\mathrm{ATHom(m,n)} = \left| x_{m,n} - \frac{x_{m-1,n} + x_{m+1,n}}{2} \right|$$

      This difference is calculated for all pixels in the image except border pixels (one pixel on all sides). Whenever a pixel in the artifact mask or a sample from an inoperable detector is encountered, ATHom should be zero, so all

pixels in the artifact mask and their top and bottom adjacent pixels will have zero values in ATHom. Thus, inoperable detectors will cause one column of zeros in ATHom.

c. To help reduce noise, we will average CTHom over five pixels.

$$\mathrm{ACTHom(m,n)} = \frac{1}{5}\begin{pmatrix}\mathrm{CTHom(m\text{-}2,n)} + \mathrm{CTHom(m\text{-}1,n)} + \\ \mathrm{CTHom(m,n)} + \mathrm{CTHom}(m+1,\mathrm{n}) + \mathrm{CTHom}(m+2,\mathrm{n})\end{pmatrix}$$

ACTHom stands for Average Cross-Track Homogenity. This is done for the entire CTHom image. Pixels in the artifact mask or a sample from an inoperable detector and their left and right adjacent pixels should not be used to calculate this average. Border pixels are averaged with their inside neighbors, this can be seen below for pixels on the left side. Pixels on the right side use similar equations.

$$\mathrm{ACTHom(1,n)} = \frac{\mathrm{CTHom(m,n)} + \mathrm{CTHom}(m+1,\mathrm{n}) + \mathrm{CTHom}(m+2,\mathrm{n})}{3}$$

$$\mathrm{ACTHom(2,n)} = \frac{\mathrm{CTHom(m\text{-}1,n)} + \mathrm{CTHom(m,n)} + \mathrm{CTHom}(m+1,\mathrm{n}) + \mathrm{CTHom}(m+2,\mathrm{n})}{4}$$

d. To reduce noise in ATHom, we will average over three pixels.

$$\mathrm{AATHom(m,n)} = \frac{\mathrm{ATHom(m,n\text{-}1)} + \mathrm{ATHom(m,n)} + \mathrm{ATHom(m,n}+1)}{3}$$

AATHom stands for Average Aross-Track Homogenity. This is done for the entire ATHom image. Pixels in the artifact mask or a sample from an inoperable detector and their top and bottom adjacent pixels should not be used to calculate this average. Border pixels are averaged with their inside neighbors, this can be seen below for pixels along the top border. Pixels along the bottom border use a similar equation.

$$\mathrm{AATHom(m,1)} = \frac{\mathrm{ATHom(m,n)} + \mathrm{ATHom(m,n}+1)}{2}$$

e. To complete the homogeneity filter, plug ACTHom and AATHom into the equation below.

$$\mathrm{HomFilt(m,n)} = \frac{1}{1 + \left(\dfrac{\dfrac{\mathrm{abs(ACTHom(m,n)} + \mathrm{AATHom(m,n))}}{2}}{\mathrm{Striping\ Metric\ Cutoff}}\right)^4}$$

HomFilt stands for Homogeneity Filter. This will generate a filter mask of roughly 1s and 0s the same size as the original image minus one border pixel from all sides. All pixels in the artifact mask and samples from inoperable detectors and left and right adjacent pixels should be zeroed out. This filter should remove scene content from the calculation of the striping metric.

5. The scene striping metric is the absolute value of the individual pixel product of the HomFilt and CTDiff. The scene striping metric shows where, spatially, in the image stripes are located. The higher the value the more striping present.

$$\text{Scene Striping Metric} = \text{abs}(\text{CTDiff}(m,n) \cdot \text{HomFilt}(m,n))$$

6. The detector striping metric is the mean of the columns of the scene striping metric. Each of the individual SCA scene striping metric arrays are concatenated to produce a single band array for the detector striping metric. This tells us how stripy a single detector is. The detector striping metric has individual values for each detector except for the first and last detectors.

   Figure 4-87 shows an example detector striping metric.

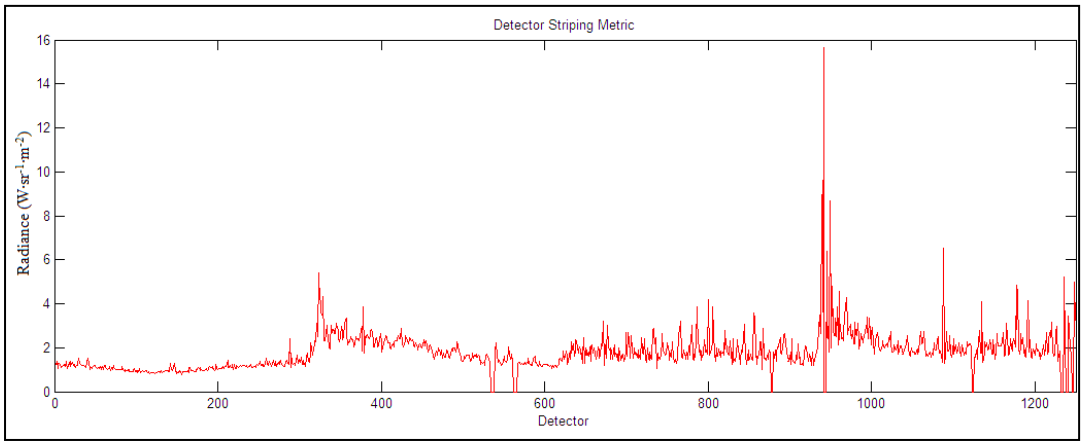

Figure 4-87. Example Detector Striping Metric

7. The overall striping metric is derived from the detector striping metric.
   a. First the mean of the entire detector striping metric is found.

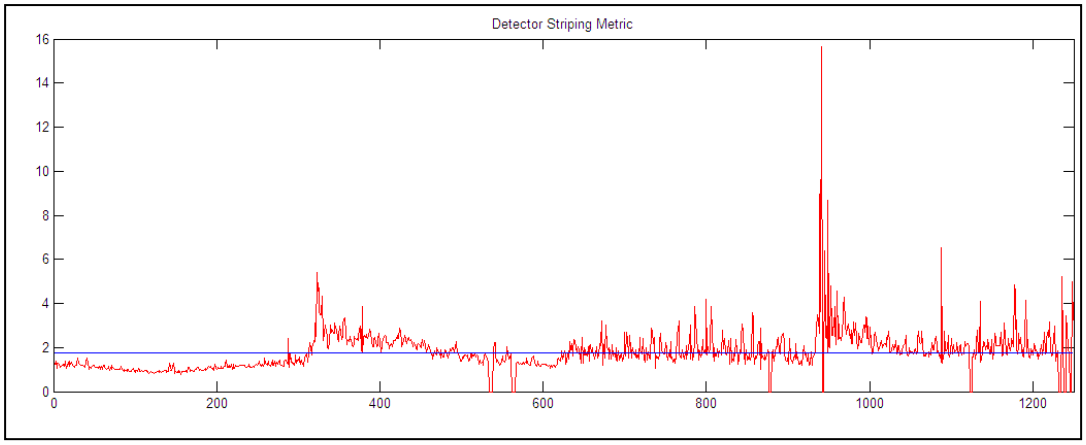

Figure 4-88. Mean of Detector Striping Metric

b. Then a 75 length median filter is applied to the detector metric, and smoothed with a 15 length averaging filter.

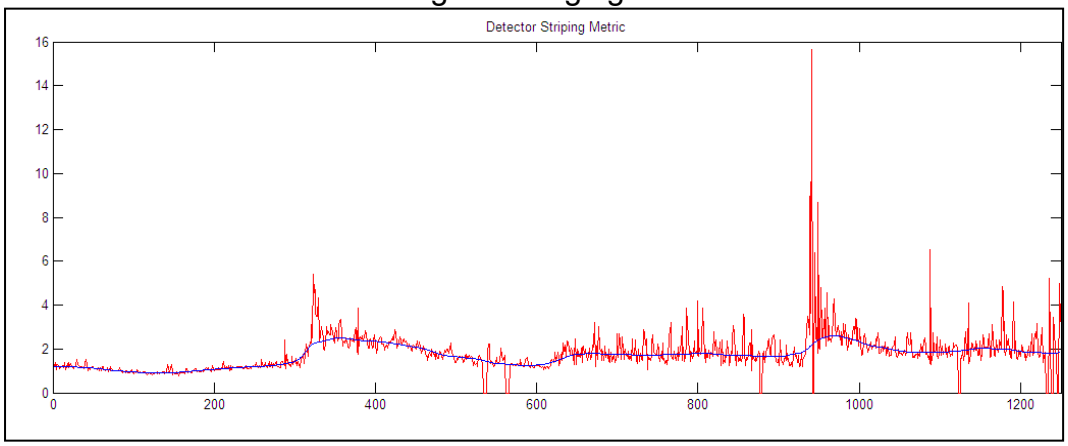

Figure 4-89. Averaged Median Fit to Detector Striping Metric

For border detectors on the left side, the median filter will find the median of 37 detectors to the right and however many detectors there are to the left. So for the first detector, it will find the median of the first detector and the 37 detectors to the right. For the second detector it will find the median of the first and second detector and the 37 detectors to the right, and so on until the 38 detector when it find the median of the current detector and 37 detectors to the left and right.

Border detectors on the right side are handled the same way except reversed. So for the last detector it will find the median of the last detector and 37 detectors to the left.

The average filter works in a similar way. For the first detector it will take the average of the first detector and 7 detectors to the right.

c. This averaged median fit is subtracted from the detector striping metric.

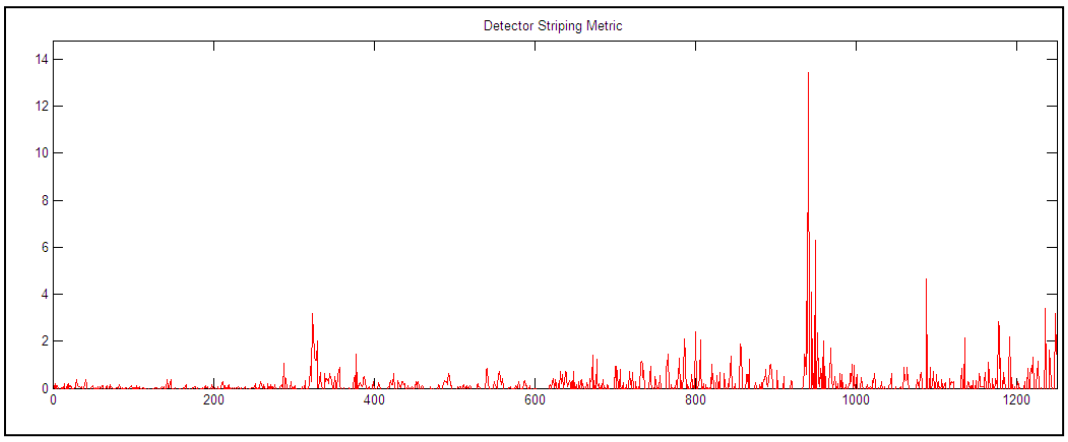

Figure 4-90. Median Subtracted Detector Striping Metric

d. The next factor used for the overall striping metric is the maximum peak from this median filter subtracted detector metric.

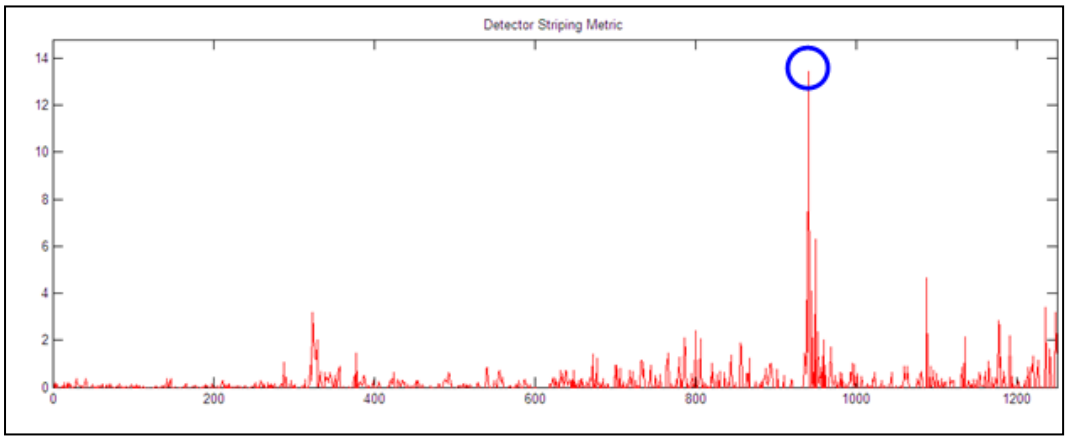

Figure 4-91. Maximum of the Median Subtracted Detector Striping Metric

e. The last factor used is then the mean of the top 15 peaks, including the maximum peak, from the median filter subtracted detector metric. (There are only six peaks circled, but the algorithm should find 15).

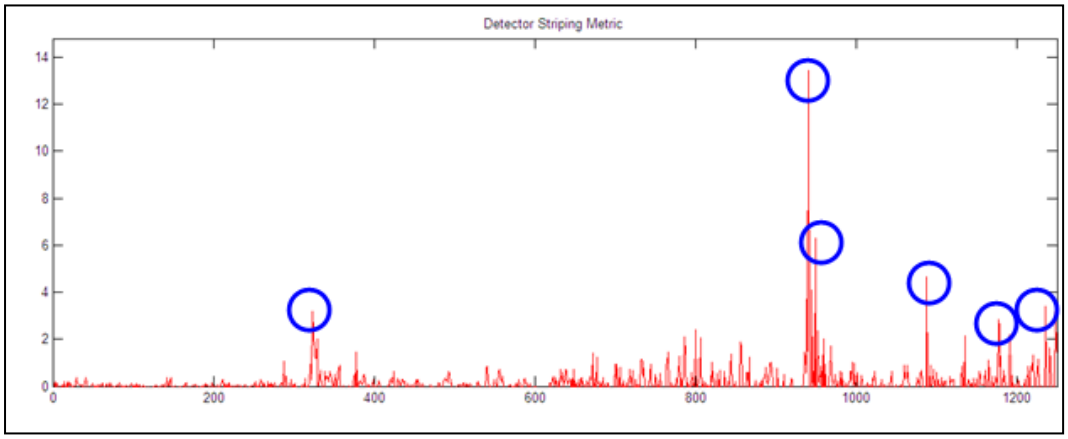

Figure 4-92. Top Peaks of the Median Subtracted Detector Striping Metric

It is important to find the top 15 individual peaks. Detectors part of a higher spike should not be used. A detector's two adjacent detectors are considered for determining peaks. If detector *x* has a neighboring detector with a higher value, detector *x* is not a peak. There is no amount a peak must be larger than its neighbors; it must only be larger. One approach to do this is to arrange the detector striping metric numbers in descending order while maintaining the detector to which the metric numbers correspond. Then one can go down the list and if there is neighboring detector above the current detector, the current detector is not an individual peak. Figure 4-93 shows this more clearly.

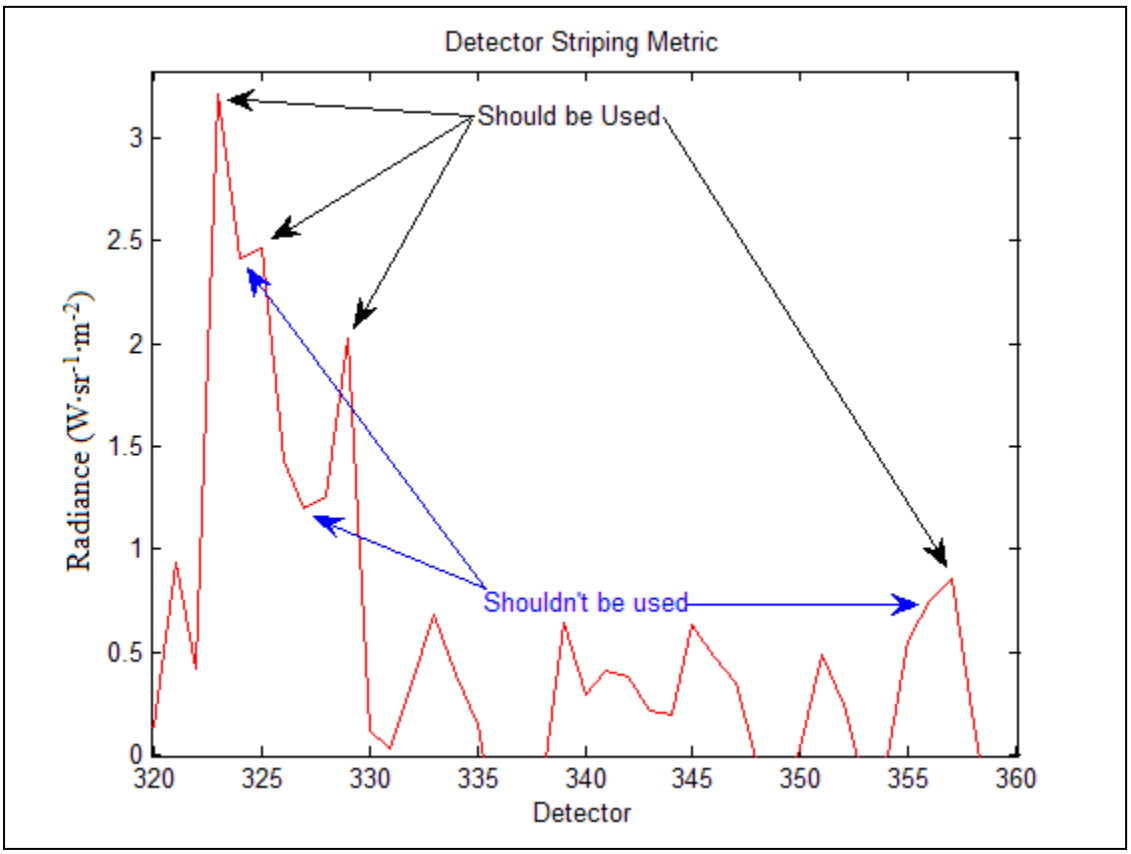

Figure 4-93. Individual Peak Detectors

f. The overall striping metric is the cube root of the product of the mean, maximum peak, and mean of the top 15 peaks. This number will be in radiance units (W·sr^{-1}·m^{-2}·µm). It is also desired to capture this value in DN, so it will have to be backed out of radiance space.

- $\text{OverallStripingMetirc(rad)} = \sqrt[3]{\text{mean} \cdot \text{max} \cdot \text{mean of top fifteen}}$
-
- $\text{OverallStripingMetric(DN)} = \text{OverallStripingMetric(rad)} * G_{abs}$

- The mean of the detector striping metric measures the amount of striping present throughout the entire image, odd/even striping being the largest portion.
- The worst single detector stripe is measured by the maximum peak.
- The mean of the top 15 peaks measures the amount of single detector striping throughout the image.

8. If the write striping metric option is on, write the overall striping metric to the database.
9. If the summary report option is on write the overall metric, scene striping metric and detector striping metric to a report.
10. Repeat all steps for all SCAs and Bands.

4.4.10 Non-uniformity Characterization

4.4.10.1 Background/Introductoin

Streaking, Banding, and Full Field Of View (FFOV) Uniformity characterizations provide 4 different measures of detector uniformity. All characterizations will generate metrics for uniformity and stability of uniformity assessments, both prelaunch and postlaunch. In the latter case, these will be characterizations will be used for Key Performance Requirement verification.

FFOV Uniformity: the standard deviation of all detector column average radiances across the FFOV within a band shall not exceed 0.5% of the average radiance

There are 2 methods for characterizing Banding.

Method A: The root mean square of the deviation from the average radiance across the full FOV for any 100 contiguous detector column averages of radiometrically corrected OLI image data within a band. This banding specification is met when the metric is less than or equal to 1.0% for OLI and 0.5% for TIRS of the band average radiance.

Method B: The standard deviation of the radiometrically corrected values across any 100 contiguous detector column averages of OLI image data within a band. This banding requirement is met when the metric is less than or equal to 0.25% for OLI and 0.5% for TIRS of the average radiance across the 100 detector columns.

Streaking is measured across any 3 contiguous detector column averages, across the FOV. The streaking requirement is met when the metric is less than 0.005 for OLI bands 1-7 and 9, and 0.01 for the OLI panchromatic band or 0.005 for the TIRS bands. The streaking parameter is defined below.

For OLI, this algorithm is intended to work primarily on solar scenes, though the capability to process uniform Earth scenes should be included; verification of the requirements at 2*Ltyp will require extra analysis. For TIRS, this algorithm should be run on blackbody scenes with a temperature set point between 260 and 330K.

These characterizations should be performed on radiance data. The Histogram Statistics Characterization module is performed on bias-corrected and linearized image data, but before the gains and relative gains have been applied. The non-uniformity characterization will not operate on the image data, but rather will use the Histogram Statistics in the database. Therefore, the algorithm will need to apply the gains from the database/CPF to convert the histogram means to radiance.

Analysis of the output data will determine whether, initially, the Non-Uniformity specification is being met and then, later, whether there are changes in uniformity across the focal plane. This algorithm requires a uniform scene or a scene of known non-uniformity. For OLI, the only target that is expected to meet this requirement is the solar diffuser. Therefore this algorithm may only be useful for characterizing the non-

uniformity performance, particularly the full field of view uniformity, on one spectral target, as opposed to the three indicated in the requirement. For TIRS, the blackbody will be operated at multiple temperature set points. Each of these blackbody images should be useful in characterizing the banding and streaking over the range of typical Earth temperatures

Note: the notation in this version of the banding equations has been modified from the OLI Requirements Document for clarification.

4.4.10.2 Inputs

The inputs to this algorithm come from either the output of other algorithms (DB) or from a set of calibration parameters (CPF).

Description	Symbol	Units	Level	Source	Type
Detector means (bias corrected, linearized only)	Q_i	DN	N_{band} x N_{SCA} x N_{det}	DB (histogram statistics table)	float
Absolute Gains	G	DN/(w/m^2 sr um)	N_{band}	CPF	float
Reference_Absolute Gains	G_{ref}	DN/(w/m^2 sr um)	N_{band}	CPF	float
SCA Relative Gains	sr_i	[]	N_{band} x N_{SCA}	CPF	float
Detector Relative gains	dr_i	[]	N_{band} x N_{SCA} x N_{det}	CPF	float
Inoperable detectors, out-of-spec detectors			N_{band} x N_{SCA} x N_{det}	CPF	integer
Solar or blackbody non-uniformity scaling factors	ν_i	[]	N_{band} x N_{SCA} x N_{det}	CPF	float

4.4.10.3 Outputs

The outputs of this algorithm are typically stored in the characterization database. However, an option to store this data to an ASCII text file is needed to support testing. This reduces inserts into the database as well as speeding up calibration updates.

Description	Symbol	Units	Level	Target	Type
BandingMetric_FFOV	B_{FFOV_i}_percent	%	N_{band} x N_{SCA} x N_{det}	Db, report	float
BandingMetric_per100pix	$B_{per100det_i}$_percent	%	N_{band} x N_{SCA} x N_{det}	Db, report	float
Full FOV Uniformity Metric	FFOV_metric_percent	%	N_{band}	Db, report	float
Streaking Metric	S_i	[]	N_{band} x N_{SCA} x N_{det}	Db, report	float

4.4.10.4 Options

- Apply non-uniformity scaling factors
- Output ASCII text file summary in addition to reporting data to the database
- Use Reference Absolute Gains instead of Absolute Gains

4.4.10.5 Procedure

For each solar collect, for each band:

1. Determine per-detector radiances (L_i') as given in Equation 1 for selected scene by applying the per-detector gains (optionally, Reference Absolute Gains) to the per-detector scene means (Q_i).

   a. $$L_i' = \frac{Q_i}{G \cdot sr_i \cdot dr_i} \quad (1)$$

      i. where *G* is the band-average gain, sr_i is the per-SCA relative gain, dr_i is the per-detector relative gain and *i* is the detector counter. In this algorithm, it is meant to count across the entire focal plane, not just across a single SCA.

2. For solar and blackbody data: Correct per-detector radiance for non-uniformity (Equation 2) of the solar panel or the blackbody using the per-detector scaling factor (ν_i). For non-solar, non-blackbody data, the scaling factors are set to 1.0 for all detectors.

   a. $$L_i = \frac{L_i'}{\nu_i} \quad (2)$$

3. Stitch the radiance data together in order across the focal plane. Include all imaging detectors. Include overlap detectors. Do not include dark or redundant detectors.

   a. Each SCA will have several detectors that image the same portion of the ground as the adjacent SCA. For example, say the SCAs each have 500 detectors and the last 10 detectors of SCA1 image the same ground as the first 10 detectors of SCA2. The radiance array should include both SCA1 detectors 491-500 and SCA2 detectors 1-10.

   b. The redundancy of the overlap detectors should not affect the banding and streaking results of the solar data though it means that we are not measuring the image SCA-to-SCA discontinuity.

4. Calculate Full FOV Uniformity Metric as given below in Equation 3 for the band. Do not include detectors flagged as inoperable or out-of-spec in the calculation.

   a. FFOV_metric_percent = stdev(L) / mean(L)*100 (3)

   b. Record FFOV_metric_percent to the database or optionally to a file.

5. Calculate banding metrics as given below in Equations 4 and 6 for each imaging detector (i). Do not include detectors flagged as inoperable or out-of-spec in the calculation for operable detectors. Skip the banding calculation for inoperable and out-of-spec detectors. Do not use inoperable or out-of-spec detectors.

a. Method 1)

$$B_{FFOV\_i} = \sqrt{\sum_{n=i}^{i+99} (L_n - \overline{L})^2 / 100} \quad (4)$$

$$B_{FFOV\_i\_percent} = \frac{B_{FFOV\_i}}{\overline{L}} * 100 \quad (5)$$

Where:

L_i is the radiance of detector i

$\overline{L}$ is the scene average radiance: $\overline{\mathrm{L}} = mean(L)$

b. Method 2)

$$B_{per100\,\mathrm{det}\_i} = \sqrt{\sum_{n=i}^{i+99} (L_n - \overline{L}_{100\,\mathrm{det}})^2 / 99} \quad (6)$$

$$B_{per100\,\mathrm{det}\_i\_percent} = \frac{B_{per100\,\mathrm{det}\_i}}{\overline{L}} * 100 \quad (7)$$

Where:

L_i is the radiance of detector i

$\overline{L}_{100\mathrm{det}}$ is the average radiance across 100 detectors

$$\overline{L}_{100\,\mathrm{det}} = \sum_{i=n}^{n+99} L_i / 100 \quad (8)$$

c. Record per-detector banding arrays to database or optionally to a file.

d. In both of these calculations, the calculation cannot be performed for the detectors at the final edge (i.e., the last 99 detectors). As a result of the banding metrics not being associated with the center detector of the window, it is really the first 50 and last 50 detectors that are not characterized. However, it is the banding entries for the last 99 detectors that are left blank.

e. In the case where i…i+99 includes inoperable and/or out-of-spec detectors, the summation should be taken for fewer detectors rather than increasing maintaining the 100 detector average.

6. Calculate streaking metric (Equation 9) for each imaging detector. Do not calculate the streaking metric for detectors flagged as inoperable and out-of-spec. Also, do not calculate streaking metric for detectors adjacent to inoperable or out-of-spec detectors.

$$S_i = \left| L_i - \frac{1}{2}\left(L_{i-1} + L_{i+1}\right) \right| / L_i \quad (9)$$

Where:

L_i is the radiance of detector i;

L_{i-1} and L_{i+1} are similarly defined for the (i-1)th and (i+1)th detector columns.

a. Record per-detector streaking array to database or optionally to a file.

4.4.11 Signal-to-Noise Characterization and Noise Equivalent delta-Temperature Characterization

4.4.11.1 Background/Introduction

Signal-to-Noise (SNR) characterization provides an estimate of the overall noise behavior of the OLI and TIRS. These noise characterizations will be performed both pre- and postlaunch. After launch, both characterizations will be used to assess instrument performance over time.

TIRS Metrics: For uniform scene temperatures between 240 K and 360 K extending over the full FOV of TIRS, and for a data collection period corresponding to a WRS-2 scene (~ 25 seconds at the nominal frame rate), the median detector standard deviation when converted into radiance units shall be ≤ 0.059 W/m^2 sr µm for the 10.8 µm channel and ≤ 0.049 W/m^2 sr µm for the 12.0 µm channel. This includes quantization noise.

The SNR is calculated on targets of per-detector uniform radiance in order to have a good estimate of the noise level at a specific radiance. For OLI the SNR characterization is performed on the diffuser panel and stim lamp collections and their associated dark collects. For TIRS, the characterization is performed on the blackbody and associated deep space collects.

Some OLI-1 lamp collects are barrel-shifted in order to monitor the lower 12-bits of the instrument. This results in saturation in the SWIR bands (SWIR1, SWIR2, and Cirrus). The results from these bands should be removed from the analysis.

These characterizations are performed on bias-corrected and linearized Histogram Statistics Characterization algorithm generated means and standard deviations (in units of digital number) stored in the IAS database; they do not require separate analyses of image data.

This algorithm calculates the SNR Characterization on a single interval and populates the SNR_Characterization table. The algorithm SNR, NEDT Trending calculates the monthly SNR/NEDT and long-term trend.

4.4.11.2 Dependencies

Histogram Statistics Characterization
Lamp Characterization

4.4.11.3 Inputs

The inputs to this algorithm come from either the output of other algorithms (DB) or from a set of calibration parameters (CPF).

Description	Symbol	Units	Level	Source	Type
calibration interval identifiers, date, time,			1	DB	

Description	Symbol	Units	Level	Source	Type
lamp pair, diffuser, truncation					
Detector means for illuminated collects (bias corrected, linearized only)	OLI: Q TIRS: Q_{HS}	DN	N_{band} x N_{SCA} x N_{det}	DB histogram statistics table or lamp response table	float
Detector means for TIRS dark collects associated with blackbody collect	TIRS: Q_0	DN	N_{band} x N_{SCA} x N_{det}	DB (histogram statistics table)	float
Detector standard deviation for illuminated collects	σ	DN	N_{band} x N_{SCA} x N_{det}	DB (histogram statistics table)	float
Detector standard deviation for paired dark collect	OLI: $\sigma_{12,0}$ TIRS: σ_0	DN	N_{band} x N_{SCA} x N_{det}	DB (histogram statistics table)	float
Number of valid frames for illuminated collect	N_{valid}	count	N_{bands} x N_{SCAs} x N_{det}	Db (histogram statistics table)	Int
Number of valid frames for paired dark collect	N_{valid_dark}	count	N_{bands} x N_{SCAs} x N_{det}	Db (histogram statistics table)	Int
gain offset (TIRS only)	gain_offset	DN	N_{band} x N_{SCA} x N_{det}	CPF	float

4.4.11.4 Outputs

The outputs of this algorithm are typically stored in the characterization database. However, an option to store this data to an ASCII text file is needed to support testing. This reduces inserts into the database as well as speeding up calibration updates.

Description	Symbol	Units	Level	Source	Type
Mean Signal Level of illuminated collect	Q	DN	N_{band} x N_{SCA} x N_{det}	DB SNR_CHARACTERIZATION	float
Signal standard deviation of illuminated collect (noise)	σ	DN	N_{band} x N_{SCA} x N_{det}	DB SNR_CHARACTERIZATION	float
Signal level of paired dark collect (assume 0 for OLI, but use real data for TIRS)	Q_0	DN	N_{band} x N_{SCA} x N_{det}	DB SNR_CHARACTERIZATION	float
Standard deviation of paired dark collect (dark noise)	σ_0	DN	N_{band} x N_{SCA} x N_{det}	DB SNR_CHARACTERIZATION	float
Number of valid frames (duplicate of histogram stat input)	N_{valid}	count	N_{band} x N_{SCA} x N_{det}	DB SNR_CHARACTERIZATION	int

Description	Symbol	Units	Level	Source	Type
Number of valid frames for paired dark collect	N_{valid_dark}	count	N_{band} x N_{SCA} x N_{det}	DB SNR_CHARACTERIZATION	Int
Noise model coefficients	a, b	[DN], [DN^2]	N_{band} x N_{SCA} x N_{det}	DB SNR_CHARACTERIZATION	float
Uncertainty in Noise model coefficients	unc_a, unc_b	[DN], [DN^2]	N_{band} x N_{SCA} x N_{det}	DB SNR_CHARACTERIZATION	float

4.4.11.5 Options

Report data to ASCII report files

4.4.11.6 Procedure

For each appropriate uniform collect, for each band:

1. Extract from Histogram Statistics database table (diffuser, blackbody) or from Lamp Response table (lamp) for each detector; Q_i, σ_i, $Q_{0,i}$ (TIRS only), $\sigma_{0,i}$, N_{valid}, and N_{valid_dark}.
    a. Lamp intervals are processed twice with two different averaging periods. The solar diffuser data are only acquired for 2 seconds, so the lamp interval averages used here should only be for 2 seconds. This algorithm should only use data from work orders where lamp_response.WARMUP_TIME_PERIOD=150
    b. For TIRS, histogram_stats happens before bias-subtraction. Need to do the bias subtraction here.
        i. $Q_i = Q_{HS,i} - Q_{0,i}$ - gain_offset
    c. For Landsat-8 OLI, filter illuminated data for acquisitions when the truncation setting was set to 1 (lower 12-bit).
        i. No filtering is required for Landsat-9 OLI as the data are all acquired at 14-bits.
    d. For Landsat-8 OLI, convert 12-bit dark noise to the equivalent 14-bit dark noise
        i. Explanation for the 1.15: "quantization noise for a quantized signal is LSB/sqrt(12), since the normal Earth imagery contains the upper 12 bits of a 14 bit number, the LSB is 4 DN"

$$\sigma_{0_i} = \sqrt{{\sigma_{12,0_i}}^2 + 1.15^2}$$

2. Calculate Signal-to-Noise ratio for each detector
    a. $SNR_i = \frac{Q_i}{\sigma_i}$ (1)

3. Calculate noise model coefficients using illuminated data and paired dark data. The equation for the noise model is $\sigma_i^2 = a + b * Q_i$

   a. NOTE: these parameters are all per-detector. Maintaining the *i* index is getting unwieldy.

   b. $a = \sigma_{0,i}^2$ (2)

   c. $b = \frac{\sigma_i^2 - a}{Q_i}$ (3)

4. Calculate uncertainty on the fit coefficients

   a. $unc_a = \frac{2a}{\sqrt{2*(N_{valid\_dark}-1)}}$ (4)

   b. $unc_{Q_i} = \frac{\sigma_i}{\sqrt{N_{valid}}}$ (5)

   c. $unc_{\sigma_i} = \frac{2\sigma_i}{\sqrt{2*(N_{valid}-1)}}$ (6)

   d. $unc_b = b * \sqrt{\left(\frac{unc_{Q_i}}{Q_i}\right)^2 + \left(\frac{\sqrt{unc_a^2 + unc_{\sigma_i}^2}}{{\sigma_i}^2 - a}\right)^2}$ (7)

5. Populate the SNR Characterization table.

4.4.12 SNR and NEDT Trending

4.4.12.1 Background/Introduction

This algorithm generates summaries of the per-interval OLI SNR and TIRS NEDT for a user-selected time period. For Landsat-8 OLI, the SNR KPRs were evaluated every month.

The SNR Characterization routine extracts the per-acquisition data necessary for generating the monthly estimate of SNR/NEDT from the histogram stat or lamp response tables and puts it into the SNR_CHARACTERIZATION table. Per-acquisition estimates of SNR are generated but these are not used for evaluating the instrument. SNR, NEDT Trending algorithm pulls the individual acquisition data for the month in question from SNR_CHARACTERIZATION and generates the monthly model for SNR. With this model, the OLI SNR at Lhigh and Ltypical can be calculated. The model coefficients are also used to calculate TIRS Noise Equivalent Detector Radiance (NEDL), which then is used to predict NEDT at the five specification temperatures.

The per-detector SNR and NEDT values are recorded to the SNR_TREND and NEDT_TREND tables. All the necessary data will be written to the tables so that the Detector Operability algorithm can determine if there are any out-of-spec detectors.

The SNR characterizations are performed on digital number (counts) stored in the IAS database. The SNR specification is written in terms of radiance, so for the final SNR

report, the SNR results need to be converted to radiance using the appropriate gains and relative gains. Similarly for TIRS data, the noise model must be converted to radiance and temperature. The temperature conversion will require use of a Planck equation approximation; a lookup table is provided for each band. The requirement radiances and temperatures are included in the SNR Requirements section.

Analysis of the output data will determine whether the SNR/NEDT requirement is being met and over time, whether there are changes across the focal plane.

4.4.12.2 Dependencies

SNR Characterization

4.4.12.3 Inputs

The inputs to this algorithm come from either the output of other algorithms (DB) or from a set of calibration parameters (CPF).

Description	Symbol	Units	Level	Source	Type
calibration interval identifiers, date, time, lamp pair, diffuser, truncation			1	DB	
start_time, end_time for trending period			1	user input	date format
OLI Spec radiance (Table 4-58)	Ltyp, Lhigh	rad	2 per band	requirements	float
OLI SNR Requirements (Table 4-59)	*spec_value*		2 per band	requirements	float
TIRS spec temperatures (Table 4-60)	T_{spec}	K	5 per band	requirements	float
TIRS NEDT Requirements (Table 4-60)	*spec_value*		5 per band	requirements	float
Detector means for illuminated collects for defined time period (bias corrected, linearized only)	Q_i	DN	$N_{acquisition}$ x N_{band} x N_{SCA} x N_{det}	DB SNR_ CHARACTERIZATION	float
Detector standard deviation for illuminated collects for the defined time period	σ_i	DN	$N_{acquisition}$ x N_{band} x N_{SCA} x N_{det}	DB SNR_ CHARACTERIZATION	float
Detector mean for paired dark collect	$Q_{0,i}$	DN	N_{band} x N_{SCA} x N_{det}	DB SNR_ CHARACTERIZATION	float
Detector standard deviation for paired dark collect for the defined time period	$\sigma_{0,i}$	DN	N_{band} x N_{SCA} x N_{det}	DB SNR_ CHARACTERIZATION	float

Description	Symbol	Units	Level	Source	Type
Absolute Gains	G	DN/(W/m^2 sr um)	N_{band}	CPF	float
SCA Relative Gains	sr_i	[]	N_{band} x N_{SCA}	CPF	float
Detector Relative gains	dr_i	[]	N_{band} x N_{SCA} x N_{det}	CPF	float
Planck curves for TIRS bands	planck	[K/radiance]	400 rows	lookup table	float

4.4.12.4 Outputs

There is a table in the database to record the output of this algorithm, however, the default is to NOT store the data.

Description	Symbol	Units	Level	Target	Type
number of collects	N_collects	[]	1	DB, report file	int
start, end date for specified period	start_date, end_date	date	1	DB, report file	date
OLI: SNR at Ltypical, Lhigh for median detector	$SNR_{Ltypical}$, SNR_{Lhigh}	[]	N_{band} x N_{SCA} x N_{det}	DB SNR_TREND, report file	float
OLI: SNR uncertainty at Ltypical, Lhigh for median detector	$unc_{SNRLtypical}$, $unc_{SNRLhigh}$	[]	N_{band} x N_{SCA} x N_{det}	DB SNR_TREND, report file	float
OLI: stdev(SNR) at Ltypical, Lhigh	σ_{SNR}	[]	N_{band} x N_{SCA} x N_{det}	DB SNR_TREND, report file	float
OLI: Noise model coefficients	a, b	[DN], [DN^2]	N_{band} x N_{SCA} x N_{det}	DB, report	float
OLI: Uncertainty in Noise model coefficients	unc_a, unc_b	[DN], [DN^2]	N_{band} x N_{SCA} x N_{det}	DB, report	float
TIRS: NEDL	NEDL	[rad]	N_{band} x N_{SCA} x N_{det}	DB NEDT_TREND, report file	float
TIRS: NEDL uncertainty	$NEDL_{unc}$	[rad]	N_{band} x N_{SCA} x N_{det}	DB NEDT_TREND, report file	float
TIRS: NEDT at spec levels (5 temperatures)	NEDT	[K]	N_{temps} x N_{band} x N_{SCA} x N_{det}	DB NEDT_TREND, report file	float
TIRS: uncertainty at spec levels	unc_{NEDT}	[K]	N_{temps} x N_{band} x N_{SCA} x N_{det}	DB NEDT_TREND report file	float

4.4.12.5 Procedure

For each user-specified period, collect the histogram statistics for the appropriate uniform collects, for each band:

1. Extract data from the SNR Characterization table for the defined time period; Q_i, σ_i, $Q_{0,i}$, $\sigma_{0,i}$. Extract absolute, SCA relative, and detector relative gains from the CPF: G, sr, and dr.
    a. From this point forward, the dark acquisitions should be treated the same at the lit acquisitions

Q_i = [Q_i, $Q_{0,i}$]

σ_i = [σ_i, $\sigma_{i,0}$]

b. For Landsat-8 OLI, filter illuminated data for collections when the truncation setting was set to 1 (lower 12-bit) (Should have already been taken care of by SNR Characterization.)

c. Count the number of valid collects

OLI: N_collects = count(n_lamps+n_solar)

TIRS: n_collects = count(n_blackbody)

2. Calculate uncertainty on the signal and noise for all levels, per-detector for all collects in the period:

a. $unc_{Q_i} = \frac{\overline{\sigma}_i}{\sqrt{2 * N_{collects}}}$

b. $unc_{\sigma_i} = \frac{2\overline{\sigma}_i}{\sqrt{2 * (N_{collects} - 1)}}$

3. Calculate noise model coefficients using all illuminated data and paired dark data in the period. The equation for the noise model is

a. [a_i, b_i] = LINFIT(Q_i,σ_i^2,sigma=unc)

b. OR in matrix terms:

$$\boldsymbol{X} = \begin{bmatrix} 1.0 & Q_{i,t0} \\ 1.0 & \dots \\ 1.0 & Q_{i,tn} \end{bmatrix}$$

$$\boldsymbol{Y} = \begin{bmatrix} \sigma_{i,to}^2 & \dots & \sigma_{i,tn}^2 \end{bmatrix}$$

$$\boldsymbol{\beta} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{Y}$$

where t0 is the first acquisition and tn is the last acquisition and the matrix includes entries from each acquisition, a = $\beta(0)$ and b = $\beta(1)$

4. Calculate uncertainty on the noise model coefficients

a. $unc_{a_i} = \frac{2a_i}{\sqrt{2 * (N_{collects} - 1)}}$

b. $unc_{b_i} = b_i * \sqrt{\left(\frac{unc_{Q_i}}{Q_i}\right)^2 + \left(\frac{\sqrt{unc_{a_i}^2 + unc_{\sigma_i}^2}}{{\sigma_i}^2 - a_i}\right)^2}$

5. For OLI:

a. Convert requirements radiances ($L_{Typical}$ and L_{High}) to per-detector digital numbers using the CPF for the scene

$$Q_{spec,i} = G * sr_i * dr_i * L_{spec_i}$$

6. For TIRS:

a. Calculate the slope and offset (m_{planck} and b_{planck}) of the Planck equation for the specification temperatures the lookup table. [requires a lookup table for each band]

i. Find the two temperatures closest to the Tspec in the lookup table. Do linear regression about those two temperatures to determine slope and offset.

b. Convert requirements spec temperatures (T_{spec}) to per-detector digital numbers using the CPF for the scene and the planck slope

$$Q_{spec,i} = \left(T_{spec} * m_{planck} + b_{planck}\right) * G * sr_i * dr_i$$

7. Calculate the uncertainties for SNR spec values using the uncertainty in the noise estimate at the spec value at the proxy for uncertainty in SNR. This results in $unc_{SNR,Lspec}$ for each detector [all parameters in should have subscript *i* but it got too tedious to add them].

a. Residual error: $\boldsymbol{E} = \boldsymbol{Y} - \boldsymbol{X\beta}$

$$s^2 = \frac{\boldsymbol{E'E}}{n-2}$$

b. use two-sided 95% t-test: $t.crit = t\_cvf(0.025, n\_collects - 2)$

c. make matrix out of spec DNs: $\boldsymbol{X_h} = \begin{bmatrix} 1.0 & Q_{spec0typ} \\ \cdots & \cdots \\ 1.0 & Q_{specnhigh} \end{bmatrix}$

d. the values we want are in the diagonal of this matrix:

$$\boldsymbol{H_x = X_h(X'X)^{-1}X'_h}$$

e. this operation should result in prediction interval (PI) for spec DN with units of noise^2 [counts^2]:

$$pi = t.crit * \sqrt{s^2 * (1 + diag(H_x))}$$

8. For OLI:

a. Calculate SNR at each spec radiance for each detector using the fit coefficients (Q_{spec} = [Q_{typ}, Q_{high}])

$$SNR_{Lspec} = \frac{Q_{spec}}{\sqrt{a_i + b_i * Q_{spec}}}$$

b. Calculate stdev of SNR across all detectors for each spec radiance

$$\sigma_{SNR_{L_{spec}}} = stdev\left(SNR_{L_{spec}}\right)$$

c. Convert the PI in noise to PI in SNR [unitless]

$$unc_{SNR,Lspec} = \frac{\sqrt{pi_{Lspec}}}{Q_{typ}} * SNR_{Lspec}$$

d. Find the detector with the median $SNR_{Ltyp,p}$ and $SNR_{Lhigh,p}$.

i. median_index_Ltyp = where(period_snr(p).snr_at_Ltyp EQ median(period_snr(p).snr_at_Ltyp))

ii. median_index_Lhigh = where(period_snr(p).snr_at_Lhigh EQ median(period_snr(p).snr_at_Lhigh))

e. Populate SNR_TREND table with per-month per-detector values.

i. Current code only writes to database if requested. I think this should be by default.

ii. if args.store_in_db:

iii. # Store the SNR Trend for the current band

iv. snr_trend.insert_row(mydb, band_number, band_snr.snr_at_ltyp[ltyp_med], band_snr.snr_at_lhigh[lhigh_med], unc_snr_at_ltyp[ltyp_med], unc_snr_at_lhigh[lhigh_med], band_snr.a[ltyp_med], band_snr.b[ltyp_med], band_snr.unc_a[ltyp_med], band_snr.unc_b[ltyp_med], used_collects, start_date, end_date, np.std(band_snr.snr_at_ltyp), np.std(band_snr.snr_at_lhigh), args.unit_test)

f. Output report files with per-period median results

g. Count the number of detectors that are less than 80% of requirement (Table 4-59), taking into account the uncertainty for each detector.

i. $out\_of\_spec\_index = where\left(SNR_{Lspec,p,i} + 2 * unc_{SNR_{Lspec,p,i}} < 0.8 * spec\_value\right)$

ii. Syntax now includes a “p” indicating per-period results, since steps 1-8 should be repeated for each user defined period.

iii. Note that while this algorithm will detect out-of-spec detectors and write them to a text report, the responsibility of counting and reporting them falls to the Detector Operability algorithm, via the data in the SNR_TREND database table.

9. For TIRS:

a. Convert specified temperature (Table 4-60) to counts:

$$Q_{Tspec} = \left(T_{spec} * m_{planck} + b_{planck}\right) * G$$

b. Calculate NEDL and NEDT and uncertainties at each specified temperature

i. $NEDL_{Tspec} = \frac{\sqrt{a+b*Q_{Tspec}}}{G*r_i}$

ii. $NEDT_{Tspec} = \frac{NEDL_{Tspec}}{m_{planck}}$

c. convert the PI in noise to PI in SNR [unitless]

i. $unc_{NEDL,Tspec} = \frac{\sqrt{pi_{Tspec}}}{Q_{Tspec}} * NEDL_{Tspec}$

ii. $unc_{NEDT,Tspec} = \frac{\sqrt{pi_{Tspec}}}{Q_{Tspec}} * NEDT_{Tspec}$

d. Find the detector with the median $NEDT_{Tspec}$ for each T_{spec}

i. median_index_Tspec = where(period_snr(p).nedt_tspec EQ median(period_snr(p).nedt_tspec))

e. Assess the NEDT at each temperature against the requirements (Table 4-60). Count and flag failing detectors, taking into account the uncertainty for each detector.

i. $out\_of\_spec\_index = where(NEDT_{Tspec,p} - 2 * unc_{NEDT,Tspec,p} > spec\_value)$

ii. Note that while this algorithm will detect out-of-spec detectors and write them to a text report, the responsibility of counting and reporting them falls to the Detector Operability algorithm, via the data in the NEDT_TREND database table.

f. Populate NEDT_TREND tables with per-month per-detector results. Output median values to report files.

10. Generate KPR-style plots. Samples shown in Figures Figure 4-94, Figure 4-95, and Figure 4-96.

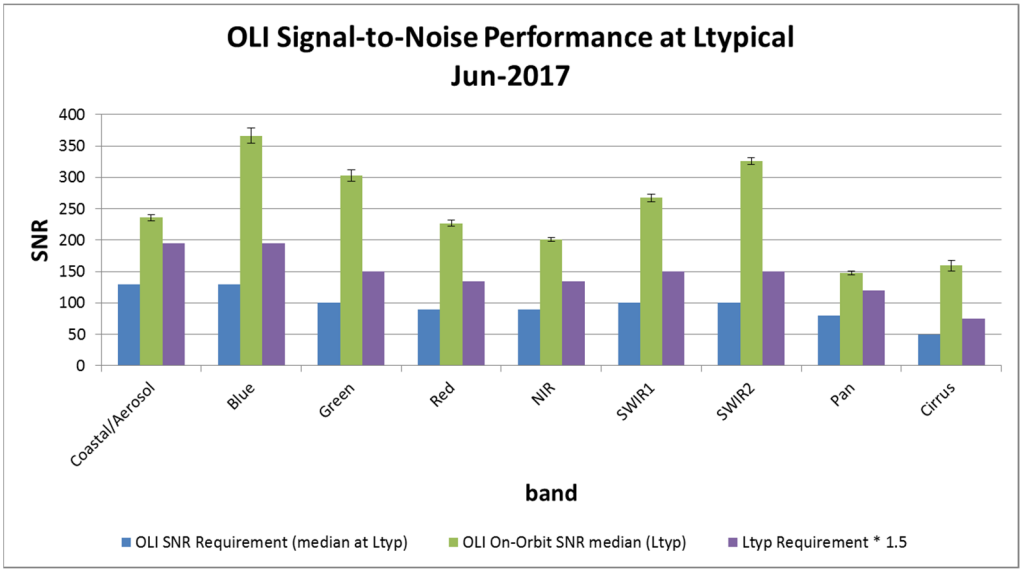

Figure 4-94. Sample OLI SNR Performance at $L_{typical}$

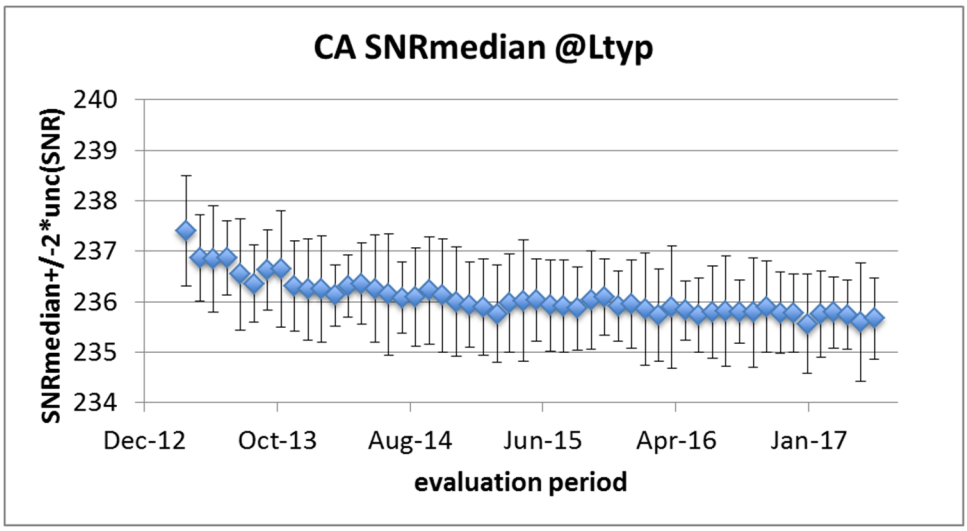

Figure 4-95. Sample OLI SNR Trend at $L_{typical}$

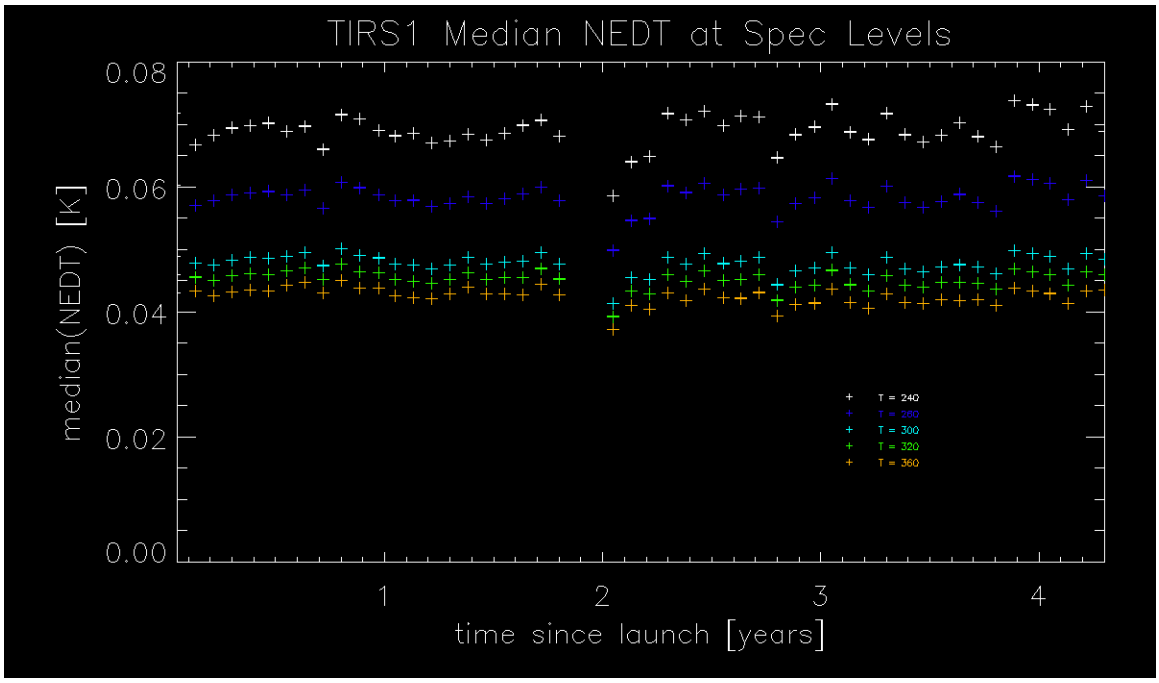

Figure 4-96. Sample TIRS SNR Trends at 5 Temperatures

4.4.12.6 SNR Requirements

#	Band	Radiance Level for SNR, L (W/m^2 sr µm)		Saturation Radiances, L_{Max} (W/m^2 sr µm)
		Typical, $L_{Typical}$	High, L_{high}	Requirement
1	Coastal Aerosol	40	190	555
2	Blue	40	190	581
3	Green	30	194	544

#	Band	Radiance Level for SNR, L (W/m^2 sr µm)		Saturation Radiances, L_{Max} (W/m^2 sr µm)
4	Red	22	150	462
5	NIR	14	150	281
6	SWIR 1	4.0	32	71.3
7	SWIR 2	1.7	11	24.3
8	Panchromatic	23	156	515
9	Cirrus	6.0	N/A	88.5

Table 4-58. Radiance Levels for SNR Requirements (from OLI Requirements Document)

#	Band	SNR Requirements	
		At $L_{Typical}$*	At L_{High}*
1	Coastal Aerosol	130	290
2	Blue	130	360
3	Green	100	390
4	Red	90	340
5	NIR	90	460
6	SWIR 1	100	540
7	SWIR 2	100	510
8	Panchromatic	80	230
9	Cirrus	50	N/A

Table 4-59. SNR Requirements

	NEDT at 240K	NEDT at 260K	NEDT at 300K	NEDT at 320K	NEDT at 360K
TIRS 1 (B10)	0.80	0.61	0.40	0.35	0.27
TIRS 2 (B11)	0.71	0.57	0.40	0.35	0.29

Table 4-60. NEDT Requirements

4.4.13 Detector Operability Characterization

4.4.13.1 Background/Introduction

This algorithm identifies anomalous detectors which either do not meet performance specifications or differ greatly from the average detector. Analysts will use the output from this algorithm to manually investigate anomalous detectors and determine whether a detector should be added to the inoperable detector list in the CPF.

A detector is considered anomalous if

1. A detector's responsivity differs from the FPM mean by Gain_Anamolous_Threshold (nominally 20%)
2. A detector's response deviates Gain_Stability_Threshold (nominally 5) times more than the FPM mean deviation.

3. A detector's dark response differs from the FPM mean by Bias_Anomalous_Threshold (nominally 20%)
4. A detector's dark response deviates Bias_Stability_Threshold (nominally 5) times more than the FPM mean deviation.
5. A detector's SNR or NEdT is less than SNR_Specification or NEdT_Specification at Specification_Radiance or Specification_Temperature, respectively, or a detector's SNR or NEdT is less than SNR_Anomalous_Threshold (nominally 20%) or NEdT_Anomalous_Threshold (nominally 20%), respectively, less than the FPM mean
6. A detector's streaking metric is greater than Streaking_Threshold (nominally 0.005 for OLI bands 1-7 and 9 and TIRS and 0.01 for the PAN band)

The exact tests for evaluating the above criteria are described in the procedure below. Detectors identified as anomalous and the specific test failed will be written to the database.

4.4.13.2 Dependencies

Relative Gain Characterization (Solar Diffuser)
Histogram Statistics Characterization
OLI Detector Response Characterization (Solar Diffuser)
OLI Detector Response Characterization (Lamp)
Signal-to-Noise Characterization
Non-Uniformity Characterization

4.4.13.3 Inputs

Description	Symbol	Units	Level	Source	Type
Calibration Interval Identifiers		N/A	1	Database	Char
CPF, BPF		N/A	1	CPF Service	Char
Inoperable Detectors		N/A	N_{band} x N_{SCA} x N_{det}	CPF	Int
Detector Gain Offset (TIRS only)	gain_offset	DN	N_{band} x N_{SCA} x N_{det}	CPF	Float
Diffuser Select		N/A	1	Diffuser_collect_type	Char
Relative_Gain_Mean	$G_{rel\ solar}$	N/A	N_{band} x N_{SCA} x N_{det}	Solar_Relgains_Detector	Float
Detector Means (lamp, shutter, blackbody, deep space)	Q_{lamp}, Q_{shutt}, Q_{black}, Q_{space}	DN	N_{band} x N_{SCA} x N_{det}	Lamp_Response_Detector or Histogram_Statistics_CAL	Float
Detector Standard Deviation (shutter, solar, blackbody, deep space)	σ_{shutt}, σ_{sol}, σ_{black}, σ_{space}	DN	N_{band} x N_{SCA} x N_{det}	Histogram_Statistics_CAL	Float
Lamp_Pair_ID		N/A	1	Lamp_Response	Char
Detector Bias (Average of Pre- and Post- Collects)	B	DN	N_{band} x N_{SCA} x N_{det}	BPF	Float
Start and End dates (for SNR Trending)		N/A	2	SNR_Trend	Date

Description	Symbol	Units	Level	Source	Type
Signal-to-noise ratio (at 2 specified radiances)	SNR_{Ltyp} SNR_{Lhigh}	N/A	2x N_{band} x N_{SCA} x N_{det}	SNR_Trend	Float
Start and End dates (for NEdT Trending)		N/A	2	NEdT_Trend	Date
Noise Equivalent delta Temperature (at 5 specified temperatures)	*NEdT*	K	5x N_{band} x N_{SCA} x N_{det}	NEdT_Trend	Float
Streaking Metric		N/A	N_{band} x N_{SCA} x N_{det}	Non-Uniformity Characterization	Float
Gain_Anomalous_Threshold		%	N_{band}	Config File/Option	Float
Gain_Stability_Threshold		N/A	N_{band}	Config File/Option	Float
Bias_Anomalous_Threshold		%	N_{band}	Config File/Option	Float
Bias_Stability_Threshold		N/A	N_{band}	Config File/Option	Float
SNR_Specification (Same as SNR Char.)	$SNR_{Ltyp\ spec}$ $SNR_{Lhigh\ spec}$	N/A	2x N_{band}	Config File/Option	Float
Specification_Radiance (Same as SNR Char.)	L_{typ} L_{high}	W/(m^2 sr µm)	2x N_{band}	Config File/Option	Float
NEdT_Specification (Same as NEdT Char.)	$NEdT_{spec}$	K	5x N_{band}	Config File/Option	Float
Specification_Temperature (Same as NEdT Char.)		K	5x N_{band}	Config File/Option	Float
SNR_Anomalous_Threshold		%	N_{band}	Config File/Option	Float
NEdT_Anomalous_Threshold		%	N_{band}	Config File/Option	Float
Streaking_Threshold		N/A	N_{band}	Config File/Option	Float

4.4.13.4 Outputs

Description	Level	Target	Type
Band, FPM, and Detector Number for each detector failing a test	3x $N_{failed\ det}$	DB	Int
Date of BPF or Interval ID of failed test	$N_{failed\ det}$	DB	Date
Number of test failed (see Table 4-61)	$N_{failed\ det}$	DB	Int
BPF or Interval ID(s) used in failed test	(2x) $N_{failed\ det}$	DB	String
Start and End Date of the SNR and/or NEdT time period used in failed test	2x $N_{failed\ det}$	DB	Date

Each failed test for each detector should generate a separate entry in the database. The date of the BPF or Interval ID should be written to the database. For the cases where two Intervals are used in one test, the date of the first Interval should be used. Not every test will have BPF or Interval ID information, so these columns may be empty. Some tests will have two Intervals, so the database should have two columns for these, which will both be empty for some tests. Tests relating to SNR or NEdT will use data from the SNR_Trend and NEdT_Trend database tables which are generated from a certain time period. The start and end dates of these time frames, which are also contained in the database tables, are needed if these tests are failed. These columns will be empty for other failed test cases.

4.4.13.5 Options

The analyst should be able to specify beginning and ending dates to evaluate tests. Each of the Config File input parameters and writing to the database should be overridable by an analyst when running this algorithm. An output report should be possible as well.

4.4.13.6 Procedure

The procedure for identifying anomalous detectors is a process of applying simple threshold tests. Each detector with a failed test will be recorded and written to the database along with the BPF or Interval Id(s) of the input used in the test. If a detector fails multiple tests, an entry will be added for each failed test.

4.4.13.6.1 Responsivity Difference

A detector's responsivity is different than the FPM mean by Gain_Anomalous_Threshold (nominally 20%).

1. A detector's relative gain within one solar diffuser collect is Gain_Anomalous_Threshold less/greater than the FPM mean
2. A detector's mean response from a blackbody collect is Gain_Anomalous_Threshold less/greater than the FPM mean.

For every solar diffuser collect within the specified time frame,
for every Band and FPM,

3. Read in Detector Relative_Gain_Mean, $G_{rel\ solar}(d)$.
4. Read in Gain_Anomalous_Threshold.
5. Ignoring inoperable detectors, calculate the FPM mean Gain, $G_{FPM\ mean}$.

$$G_{FPM\ mean} = \overline{G_{rel\ solar}(d)} \quad (1)$$

6. Identify every detector with a relative gain, $G_{rel\ solar}(d)$, less than (1 - Gain_Anomalous_Threshold) * $G_{FPM\ mean}$ or greater than (1 + Gain_Anomalous_Threshold) * $G_{FPM\ mean}$. (Test Identifier = 1)

For every blackbody collect within the specified time frame,
for every Band and FPM,

1. Read in Detector means from the blackbody collect, $Q_{black}(d)$.
2. Read in Detector means from the deep space collect, $Q_{space}(d)$, paired (within 0.003 days) with the blackbody collect.
3. Read in Detector gain offsets, $gain\_offset(d)$.
4. Read in Gain_Anomalous_Threshold.
5. Subtract bias and gain_offsets.

$$Q'_{black}(d) = Q_{black}(d) - Q_{space}(d) - gain\_offset(d) \quad (2)$$

6. Ignoring inoperable detectors, calculate the FPM mean, $Q_{FPM\ mean}$.

$$Q_{FPM\ mean} = \overline{Q'_{black}(d)} \tag{3}$$

7. Identify every detector with a mean, $Q'_{black}(d)$,
   less than (1 - Gain_Anomalous_Threshold) * $Q_{FPM\ mean}$
   or greater than (1 + Gain_Anomalous_Threshold) * $Q_{FPM\ mean}$.
   (Test Identifier = 2)

4.4.13.6.2 Response Stability

A detector's gain deviates Gain_Stability_Threshold (nominally 5) times more than the FPM mean deviation.

1. A detector's standard deviation from one solar diffuser or blackbody collect is Gain_Stability_Threshold times larger than the FPM mean detector standard deviation.

For every solar diffuser collect within the specified time frame,
for every Band and FPM,
1. Read in Detector standard deviations, $\sigma_{sol}(d)$.
2. Read in Gain_Stability_Threshold.
3. Ignoring inoperable detectors, calculate FPM mean standard deviation, $\sigma_{FPM\ mean}$.

$$\sigma_{FPM\ mean} = \overline{\sigma_{sol}(d)} \tag{4}$$

4. Identify every detector with a standard deviation, $\sigma_{sol}(d)$,
   greater than Gain_Stability_Threshold * $\sigma_{FPM\ mean}$.
   (Test Identifier = 3)

For every blackbody collect within the specified time frame,
for every Band and FPM,
1. Read in Detector standard deviations, $\sigma_{black}(d)$.
2. Read in Gain_Stability_Threshold.
3. Ignoring inoperable detectors, calculate FPM mean standard deviation, $\sigma_{FPM\ mean}$.

$$\sigma_{FPM\ mean} = \overline{\sigma_{black}(d)} \tag{5}$$

4. Identify every detector with a standard deviation, $\sigma_{black}(d)$,
   greater than Gain_Stability_Threshold * $\sigma_{FPM\ mean}$.
   (Test Identifier = 4)

2. A detector's mean responses from two consecutive solar diffuser, lamp, or blackbody collects differ more than Gain_Stability_Threshold times the FPM mean detector responses difference.

For every consecutive working solar diffuser collect pair with nominal integration time within the specified time frame,

for every Band and FPM,

1. Read in Detector Relative_Gain_Means from two consecutive collects, $G_{1\ rel\ solar}(d)$ and $G_{2\ rel\ solar}(d)$.
2. Read in Gain_Stability_Threshold.
3. Subtract the detector Relative_Gain_Means.

$$RG_{diff}(d) = abs(G_{1\ rel\ solar}(d) - G_{2\ rel\ solar}(d)) \quad (6)$$

4. Ignoring inoperable detectors, calculate the FPM mean difference, $RG_{FPM\ diff\ mean}$.

$$RG_{FPM\ diff\ mean} = \overline{RG_{diff}(d)} \quad (7)$$

5. Identify every detector with a relative gain difference, $RG_{diff}(d)$, greater than Gain_Stability_Threshold * $RG_{FPM\ diff\ mean}$. (Test Identifier = 5)

For every consecutive lamp collect pair within the specified time frame (Consecutive here is in reference to a series of collects with the same operating parameters (i.e. Lamp_Pair_ID and, for L8, lower or upper 12-bit transmission), so two lamps collects with the same Lamp_Pair_ID and bit transmission may be considered consecutive if another lamp collect with a different Lamp_Pair_ID or bit transmission setting was acquired between them. Another issue, with L8, with lower 12-bit lamp collects is a possible saturation effect in the SWIR1, SWIR2, and Cirrus bands. These bands should be excluded in the analysis of lower 12-bit collects.),

for every Band and FPM,

1. Read in Detector means, from the 150 second warmup time, from two consecutive collects, $Q_{1\ lamp}(d)$ and $Q_{2lamp}(d)$.
2. Read in Gain_Stability_Threshold.
3. Subtract the detector means.

$$Q_{diff}(d) = Q_{1\ lamp}(d) - Q_{2\ lamp}(d) \quad (8)$$

4. Ignoring inoperable detectors, calculate the FPM mean difference, $Q_{FPM\ diff\ mean}$.

$$Q_{FPM\ diff\ mean} = \overline{Q_{diff}(d)} \quad (9)$$

5. Subtract the FPM mean difference, $Q_{FPM\ diff\ mean}$, from the individual detector differences, $Q_{diff}(d)$.

$$Q_{abs\ diff}(d) = abs(Q_{diff}(d) - Q_{FPM\ diff\ mean}) \quad (10)$$

6. Ignoring inoperable detectors, calculate the FPM mean absolute difference, $Q_{FPM\ abs\ diff}$.

$$Q_{FPM\ abs\ diff} = \overline{Q_{abs\ diff}(d)} \quad (11)$$

7. Identify every detector with an absolute difference, $Q_{abs\ diff}(d)$, greater than Gain_Stability_Threshold * $Q_{FPM\ abs\ diff}$. (Test Identifier = 6)

For every consecutive blackbody collect pair within the specified time frame,
for every Band and FPM,

1. Read in Detector means from the two consecutive blackbody collects, $Q_{1black}(d)$ and $Q_{2\ black}(d)$.
2. Read in Detector means for the deep space collect(s), $Q_{1\ space}(d)$ and $Q_{2\ space}(d)$, paired (within 0.003 days) with the blackbody collects.
3. Read in Detector gain offsets for the blackbody collects, $gain\_offset_1(d)$ and $gain\_offset_2(d)$.
4. Read in Gain_Stability_Threshold.
5. Subtract bias and gain_offsets.

$$Q'_{1\ black}(d) = Q_{1\ black}(d) - Q_{1\ space}(d) - gain\_offset_1(d) \quad (12)$$
$$Q'_{2\ black}(d) = Q_{2\ black}(d) - Q_{2\ space}(d) - gain\_offset_2(d) \quad (13)$$

6. Subtract the detector means.

$$Q_{diff}(d) = Q'_{1\ black}(d) - Q'_{2\ black}(d) \quad (14)$$

7. Ignoring inoperable detectors, calculate the FPM mean difference, $Q_{FPM\ diff\ mean}$.

$$Q_{FPM\ diff\ mean} = \overline{Q_{diff}(d)} \quad (15)$$

8. Subtract the FPM mean difference, $Q_{FPM\ diff\ mean}$, from the individual detector differences, $Q_{diff}(d)$.

$$Q_{abs\ diff}(d) = abs(Q_{diff}(d) - Q_{FPM\ diff\ mean}) \quad (16)$$

9. Ignoring inoperable detectors, calculate the FPM mean absolute difference, $Q_{FPM\ abs\ diff}$.

$$Q_{FPM\ abs\ diff} = \overline{Q_{abs\ diff}(d)} \quad (17)$$

10. Identify every detector with an absolute difference, $Q_{abs\ diff}(d)$, greater than Gain_Stability_Threshold * $Q_{FPM\ abs\ diff}$. (Test Identifier = 7)

4.4.13.6.3 Dark Response Difference

A detector's dark response differs from the FPM mean by Bias_Anomalous_Threshold (nominally 20%).

1. A detector's bias is Bias_Anomalous_Threshold less/greater than the FPM mean.

For every BPF with effective dates within the specified time frame,
for every Band and FPM,
1. Read in Detector Biases, *B(d)*.
2. Read in Bias_Anomalous_Threshold.
3. Ignoring inoperable detectors, calculate the FPM mean Bias, $B_{FPM\ mean}$.

$$B_{FPM\ mean} = \overline{B(d)} \tag{18}$$

4. Identify every detector with a bias, *B(d)*,
less than (1 - Bias_Anomalous_Threshold) * $B_{FPM\ mean}$
or greater than (1 + Bias_Anomalous_Threshold) * $B_{FPM\ mean}$.
(Test Identifier = 8)

2. A detector's mean response from a shutter or deep space collect is Bias_Anomalous_Threshold less/greater than the FPM mean.

For every shutter collect (greater than 1.5s and less than 5s) within the specified time frame,
for every Band and FPM,
1. Read in Detector means, $Q_{shutt}(d)$.
2. Read in Bias_Anomalous_Threshold.
3. Ignoring inoperable detectors, calculate the FPM mean, $Q_{FPM\ mean}$.

$$Q_{FPM\ mean} = \overline{Q_{shutt}(d)} \tag{19}$$

4. Identify every detector with a mean, $Q_{shutt}(d)$,
less than (1 - Bias_Anomalous_Threshold) * $Q_{FPM\ mean}$
or greater than (1 + Bias_Anomalous_Threshold) * $Q_{FPM\ mean}$.
(Test Identifier = 9)

For every deep space collect within the specified time frame,
for every Band and FPM,
1. Read in Detector means, $Q_{space}(d)$.
2. Read in Bias_Anomalous_Threshold.
3. Ignoring inoperable detectors, calculate the FPM mean, $Q_{FPM\ mean}$.

$$Q_{FPM\ mean} = \overline{Q_{space}(d)} \tag{20}$$

4. Identify every detector with a mean, $Q_{space}(d)$,

less than (1 - Bias_Anomalous_Threshold) * $Q_{FPM\ mean}$
or greater than (1 + Bias_Anomalous_Threshold) * $Q_{FPM\ mean}$.
(Test Identifier = 10)

4.4.13.6.4 Dark Response Stability

A detector's dark response deviates Bias_Stability_Threshold (nominally 5) times more than the FPM mean deviation.

1. A detector's standard deviation from one shutter or deep space collect is Bias_Stability_Threshold times larger than the FPM mean detector standard deviation.

For every shutter collect (greater than 1.5s and less than 5s) within the specified time frame,
for every Band and FPM,
1. Read in Detector standard deviations, $\sigma_{shutt}(d)$.
2. Read in the Bias_Stability_Threshold.
3. Ignoring inoperable detectors, calculate FPM mean standard deviation, $\sigma_{FPM\ mean}$.

$$\sigma_{FPM\ mean} = \overline{\sigma_{shutt}(d)} \tag{21}$$

4. Identify every detector with a standard deviation, $\sigma_{shutt}(d)$,
greater than Bias_Stability_Threshold * $\sigma_{FPM\ mean}$.
(Test Identifier = 11)

For every deep space collect within the specified time frame,
for every Band and FPM,
1. Read in Detector standard deviations, $\sigma_{space}(d)$.
2. Read in the Bias_Stability_Threshold.
3. Ignoring inoperable detectors, calculate FPM mean standard deviation, $\sigma_{FPM\ mean}$.

$$\sigma_{FPM\ mean} = \overline{\sigma_{space}(d)} \tag{22}$$

4. Identify every detector with a standard deviation, $\sigma_{space}(d)$,
greater than Bias_Stability_Threshold * $\sigma_{FPM\ mean}$.
(Test Identifier = 12)

2. A detector's mean responses from two consecutive shutter or deep space collects differ more than Bias_Stablility_Threshold times the FPM mean detector responses difference.

For every shutter collect pair (greater than 1.5s and less than 5s) within the specified time frame,
for every Band and FPM,

1. Read in Detector means from two consecutive collects, $Q_{1\ shutt}(d)$ and $Q_{2shutt}(d)$.
2. Read in the Bias_Stability_Threshold.
3. Subtract the detector means.

$$Q_{diff}(d) = abs(Q_{1\ shutt}(d) - Q_{2\ shutt}(d)) \quad (23)$$

4. Ignoring inoperable detectors, calculate the FPM mean difference, $Q_{FPM\ diff\ mean}$.

$$Q_{FPM\ diff\ mean} = \overline{Q_{diff}(d)} \quad (24)$$

5. Identify every detector with a difference, $Q_{diff}(d)$, greater than Bias_Stability_Threshold * $Q_{FPM\ diff\ mean}$. (Test Identifier = 13)

For every deep space collect pair within the specified time frame,
for every Band and FPM,

1. Read in Detector means from two consecutive collects, $Q_{1\ space}(d)$ and $Q_{2space}(d)$.
2. Read in the Bias_Stability_Threshold.
3. Subtract the detector means.

$$Q_{diff}(d) = abs(Q_{1\ space}(d) - Q_{2\ space}(d)) \quad (25)$$

4. Ignoring inoperable detectors, calculate the FPM mean difference, $Q_{FPM\ diff\ mean}$.

$$Q_{FPM\ diff\ mean} = \overline{Q_{diff}(d)} \quad (26)$$

5. Identify every detector with a difference, $Q_{diff}(d)$, greater than Bias_Stability_Threshold * $Q_{FPM\ diff\ mean}$. (Test Identifier = 14)

4.4.13.6.5 SNR and NEdT

A detector's SNR or NEdT is less than SNR_Specification or NEdT_Specification at Specification_Radiance or Specification_Temperature, respectively, or a detector's SNR or NEdT is less than SNR_Anomalous_Threshold (nominally 20%) or NEdT_Anomalous_Threshold (nominally 20%), respectively, less than the FPM mean.

1. A detector's SNR or NEdT is less than SNR_Specification or NEdT_Specification at Specification_Radiance or Specification_Temperature.
2. A detector's SNR from a solar diffuser, lamp, or blackbody collect is SNR_Anomalous_Threshold or NEdT_Anomalous_Threshold less than the FPM mean.

For every SNR entry overlapping with the specified time frame,
for every Band and FPM,

1. Read in Detector SNRs for all specified radiances, $SNR_{Ltyp}(d)$ and $SNR_{Lhigh}(d)$.
2. Read in SNR_Specifications, $SNR_{Ltyp\ spec}$ and $SNR_{Lhigh\ spec}$.

3. Identify every detector with an SNR, $SNR_{Ltyp}(d)$ or $SNR_{Lhigh}(d)$, less than SNR_Specification, $SNR_{Ltyp\ spec}$ or $SNR_{Lhigh\ spec}$. (Test Identifier = 15 for L_{typ} and 16 for L_{high})
4. Ignoring inoperable detectors, calculate FPM mean SNR at each specified radiance, $SNR_{FPM\ mean\ Ltyp}$ and $SNR_{FPM\ mean\ Lhigh}$.

$$SNR_{FPM\ mean\ Ltyp} = \overline{SNR_{Ltyp}(d)} \tag{27}$$

$$SNR_{FPM\ mean\ Lhigh} = \overline{SNR_{Lhigh}(d)} \tag{28}$$

5. Identify every detector with an SNR, $SNR_{Ltyp}(d)$ or $SNR_{Lhigh}(d)$, less than (1 – SNR_Anomalous_Threshold) * FPM mean SNR, $SNR_{FPM\ mean\ Ltyp}$ or $SNR_{FPM\ mean\ Lhigh}$. (Test Identifier = 17 for L_{typ} and 18 for L_{high})

For every NEdT entry overlapping with the specified time frame,
for every Band and FPM,

1. Read in Detector NedTs for all specified temperatures, $NEdT(d)$.
2. Read in NEdT_Specifications, $NEdT_{spec}$.

3. Identify every detector with an NEdT, $NEdT(d)$, less than NEdT_Specification, $NEdT_{spec}$. (Test Identifier = 19, 20, 21, 22, and 23 for each specification temp)
4. Ignoring inoperable detectors, calculate FPM mean NEdT at each specified temperature, $NEdT_{FPM\ mean}$.

$$NEdT_{FPM\ mean} = \overline{NEdT(d)} \tag{29}$$

5. Identify every detector with an NEdT, $NEdT(d)$, less than (1 – NEdT_Anomalous_Threshold) * $NEdT_{FPM\ mean}$. (Test Identifier = 24, 25, 26, 27, and 28 for each specification temp)

4.4.13.6.6 Streaking

A detector's streaking metric is greater than Streaking_Threshold (nominally 0.005 for OLI bands 1-7 and 9 and TIRS and 0.01 for the PAN band).

1. A detector's streaking metric from a solar diffuser or blackbody collect is greater than the Streaking_Threshold.

For every solar diffuser collect

for every Band and FPM,

1. Read in Detector streaking metrics.
2. Read in Streaking_Threshold.
3. Identify every detector with a streaking metric greater than the Streaking_Threshold.
   (Test Identifier = 29)

For every blackbody collect

for every Band and FPM,

1. Read in Detector streaking metrics.
2. Read in Streaking_Threshold.
3. Identify every detector with a streaking metric greater than the Streaking_Threshold.
   (Test Identifier = 30)

4.4.13.6.7 Write to Database

1. Write Band, FPM, and Detector number for all detectors failing at least one test.
2. Write the identifying number of the test failed. Table 4-61 contains a list of each test with its identifying number.
3. Write the BPF name, Calibration Interval ID(s), or SNR/NEdT start and end dates used in the failed test.

Only detectors which fail a test will have an entry written to the database. A different entry will be written for each failed test. This algorithm can be run over the same time period or overlapping time periods multiple times. Duplicate entries are not of value, but the newest results should be reflected in the database, so older duplicate entries should be overwritten.

4.4.13.6.8 Output Report (Optional)

The optional output report should include the number of total tests failed and the number of detectors which failed at least one test. A full list of detectors which failed at least one test with the failed test number(s) should also be included. This report should include all instances of failed tests during the time period even if they were previously in the database. Finally, the time frame the algorithm was run over is needed.

4.4.13.7 Test Identifier Table

Metric	Test Number Identifier
Solar Diffuser Responsivity Difference	1
Blackbody Responsivity Difference	2
Solar Diffuser Single Collect Response Stability	3
Blackbody Single Collect Response Stability	4
Solar Diffuser Consecutive Collect Response Stability	5

Metric	Test Number Identifier
Lamp Consecutive Collect Response Stability	6
Blackbody Consecutive Collect Response Stability	7
BPF Dark Response Difference	8
Shutter Dark Response Difference	9
Deep Space Dark Response Difference	10
Shutter Single Collect Dark Response Stability	11
Deep Space Single Collect Dark Response Stability	12
Shutter Consecutive Collect Dark Response Stability	13
Shutter Consecutive Collect Dark Response Stability	14
Solar Diffuser SNR Specification	15 and 16
Solar Diffuser SNR Difference	17 and 18
Blackbody NEdT Specification	19-23
Blackbody NEdT Difference	24-28
Solar Diffuser Streaking	29
Blackbody Streaking	30

Table 4-61. Test Identifier Table

4.4.14 SCA Overlap Statistics Characterization

4.4.14.1 Background/Introduction

This algorithm computes statistics information about instrument response to earth scenes in the overlap zones between SCAs for both OLI and TIRS sensors. This algorithm supports TIRS ghosting correction analysis as well as trending OLI full FOV uniformity and residual non-linearity gain errors on levels below 0.5%. Results are stored in database initially for trending and analysis that later are will be used in deriving calibration parameters corrections. This algorithm can be used on any earth scene (day or night), TIRS and OLI combined collects or collects that have data only for one payload. This algorithm will run in both LPGS and IAS.

The input provided at minimum will be a set of two arrays of radiometrically corrected data holding the responses for the right side and left side of co-registered pixels pairs for each overlap zone and every band. The array could be subsampled in the along path direction (to mimic L1R) therefore the resultant length of the input array will depend on the sub sampling in the matrix column from the existing pairs. This number may vary by band and zone , therefore the number of sub-sampled pairs per zone and band is also provided along with the arrays. This input will be produced directly by using the resampler output and pre-processing it via the data ingest steps. Additional inputs are 5 more parameters about the general scene input id, current FPM relative gains used to

process the scene and estimates for the indices for line and sample in L1T product and in L1R sources from each module in the overlap.

The algorithm outputs are a maximum set of 31 parameters out of which 23 are stored for standard LPGS processing. The parameters are computed and will be stored into the database for each SCA overlap region and for each band. Database will be used by sub-setting the data with any combinations of the following attributes: WRS-2 Path/Row, date, signal level in overlap, sun azimuth, FPA temperature, FPA window temperature. The parameters stored are: The mean and standard deviations of response for the left most module in each zone, the response ratio mean and standard deviation, the response difference mean and standard deviation, rough quick view estimate for Overlap statistics derived FPM relative gains and a related band average normalization factor per band, along with a set of parameters that characterize the along track variation to the response ratio and response difference. Those along track parameters will involve 4 parameters for a 3rd order polynomial fit to the along track smoothed response, the maximum level to the fit residual, and quality of fit parameters that include, the number of data points used to produce the fit, number of points where along track residual is larger than 3-sigma, fit success flag, RMS error of fit, Chi-Square, and lastly the Pearson correlation between fit and actual data. Optional outputs that will be available for debugging the algorithm will be a temporary file that holds the per line along-track smoothed sca-overlap ratio and sca-overlap difference per overlap zone, per band, as well as exporting into a file upto to 8 input parameters that are provided by the LPGS resampler.

For deriving from this database a best estimate for overlap statistics derived FPM relative gains requires collating the results from many datasets at a specific range of signals from which the mean ratio and mean differences in signals are computed for every overlap zone. Then those “uniform” signal estimates can be used to compute the FPM relative gains in the similar manner to the procedure described here. This way effects error included by residual non-linearity and non-uniformity of signals across the full filed of view are circumvented.

4.4.14.2 Inputs

For every co-registered point “n” in the L1T product of Scene ID - ### provide the following (over 200,000 points per boundary are possible)

Description	Symbol	Units	Level	IAS only	Source
Band b Input L1R at SCA boundary zone z {1..13} from Left SCA contributing to the L1T data position n in overlap zone (from SCA(z) for OLI or SCA 4-z for TIRS)	$Q^{L}_{(b,z,n^*)}$	Scaled integer or Float [w/m2 sr um]	N_{band} x (N_{SCA-1}) x N_n	N	Special output file from GPS resampler
Band b Input L1R at SCA boundary zone z {1..13} from Right SCA contributing to the L1T	$Q^{R}_{(b,z,n^*)}$	Scaled integer or Float [w/m2 sr um]	N_{band} x (N_{SCA-1}) x N_n	N	Special output file from GPS resampler

Description	Symbol	Units	Level	IAS only	Source
data position n in overlap zone (from SCA(z+1) for OLI or from SCA(3-z) for TIRS)					
Number of co-registered pixel pairs for zone z and band b (with which ever subsampling used to produce $Q^{L}_{(b,z,n^*)}$ and $Q^{R}_{(b,z,n^*)}$)	$N_{(b,z)}$	unitless	N_{band} x (N_{SCA-1})	N	Special output file from GPS resampler
General data for report	Scene ID , Ancillary temp data ,etc…	Various, string, etc…		Y	L0R
Line & sample coordinated for sample n in L1T	$x^{L1T}_{(b,z,n)}, y^{L1T}_{(b,z,n)}$	Integer	N_{band} x (N_{SCA-1}) x N_n	Y	Special output file from GPS resampler
Line & sample coordinated for sample n in L1R from left SCA in boundary zone z (from SCA(z) for OLI or SCA 4-z for TIRS)	$x^{L-L1R}_{(b,z,n)}, y^{L-L1R}_{(b,z,n)}$	Float	N_{band} x (N_{SCA-1}) x N_n	Sample – Y Line – N	Special output file from GPS resampler
Line & sample coordinated for sample n in L1R from right SCA in boundary zone z (from SCA(z+1) for OLI or from SCA(3-z) for TIRS)	$x^{R-L1R}_{(b,z,n)}, y^{R-L1R}_{(b,z,n)}$	Float	N_{band} x (N_{SCA-1}) x N_n	Sample – Y Line – N	Special output file from GPS resampler
L1Gsys file location	Filename1	string		Y	Optional
FPM relative gains	$G_{FPM\_rel\ (b,N_{SCA})}$	Float	N_{band} x N_{SCA}	N	CPF or RPS Cache

Co-registered pixel pairs in each overlap zone N_n {1..N)

Overlap zones indexing would be consistent in the Database across datasets. Meaning that the zone numbering indicates an across track progression which is maintained in the same direction for both TIRS and OLI. As defined in the descriptions for variables in the tables above or below it designed from looking at daytime OLI and TIRS scenes. As defined here an increase in overlap zone moves from west side to East side of the scene for both OLI and TIRS. Regardless if the scenes are Day scenes or Night scenes overlap zone-indexing definitions is tied to the same SCAs. Resampler outputs will automatically produce a consistent overlap zone indexing for all scene types.

4.4.14.3 Outputs

Description	Symbol	Units	Level	IAS only	Target
Number of data points in overlap used in computations (after filtering along track data only)	$N_{(b,z)}$	Long value will be <500,000	N_{band} x (N_{SCA-1})	N	DB [SCA overlap data]
Ratio mean	$r\_mean_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB (Gains & Solar L1R Stability)
Ratio Stdev	$r\_sd_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB (Gains & Solar L1R Stability)
Along track Ratio 4 parameters of 3rd order polynomial fit	$r\_fit_{(b,z,4)}$	Float	N_{band} x (N_{SCA-1}) x 4	N	DB
Ratio - maximum level of fit residual	$r\_\max/resid_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB
Ratio – Chi-Square	$r\_chisqr_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB
Ratio – Pearson correlation coeff between fit and data	$r\_R-factor_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB
Ratio – fit success flag	$r\_Fit\_flag_{(b,z)}$	Integer	N_{band} x (N_{SCA-1})	N	DB
Ratio – RMSE fit error as Stdev of residual	$r\_RMSE_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB
Ratio – number of points with residual greater than 3-sigma	$r\_N\_above_{(b,z)}$	long	N_{band} x (N_{SCA-1})	N	DB
Difference mean	$d\_mean_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB
Difference Stdev	$d\_sd_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB
Difference - 4 parameters of 3rd order polynomial fit along track	$d\_fit_{(b,z,4)}$	Float	N_{band} x (N_{SCA-1}) x 4	N	DB
Difference - maximum level of fit residual	$d\_\max/resid_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB
Difference – Chi-Square	$d\_chisqr_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB
Difference – Pearson correlation coeff between fit and data	$d\_R-factor_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB

Description	Symbol	Units	Level	IAS only	Target
Difference – fit success flag	$d\_Fit\_flag_{(b,z)}$	Integer	N_Bx(N_{SCA}-1)	N	DB
Difference – RMSE fit error as Stdev of residual	$d\_RMSE_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB
Difference – number of points with residual greater than 3-sigma	$d\_N\_above_{(b,z)}$	Integer	N_{band} x (N_{SCA-1})	N	DB
FPM segment Signal mean	$Q^L\_mean_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB
FPM segment Signal Standard Deviation	$Q^L\_sd_{(b,z)}$	Float	N_{band} x (N_{SCA-1})	N	DB
Overlap Statistics Derived FPM_Relative gains (a quick rough estimate)	$OSD\_G_{FPM\_rel\,(b,N_{SCA})}$	Float	N_{band} x N_{SCA}	N	DB
Correction band average Normalization factor	mean_SCA_cf(b)	Float	N_{band}	N	DB
Source type	S_Type	integer	[0 – L1G_sys] [1 – LPGS input]	Y	ODL log
Smoothed along-track per band per zone per line overlap zones response ratio. For TIRS and OLI	Rs_ratio(b,z,n*)	float	N_{band} x (N_{SCA-1}) x $N_{(b,z)}$	Y	Temporary file
Smoothed along-track per band per zone per line overlap zones response differences. For TIRS and OLI	Rs_delta(b,z,n*)	float	N_{band} x (N_{SCA-1}) x $N_{(b,z)}$	Y	Temporary file
Band b Input L1R at SCA boundary zone z {1..13} from Left SCA contributing to the L1T data position n in overlap zone (from SCA(z) for OLI or SCA 4-z for TIRS)	$Q^L_{(b,z,n^*)}$	Scaled integer or Float [w/m2 sr um]	N_{band} x (N_{SCA-1}) x N_n	Y	Temporary file
Band b Input L1R at SCA boundary zone z {1..13} from Right SCA contributing to the L1T data position n in overlap zone (from SCA(z+1) for OLI or from SCA(3-z) for TIRS)	$Q^R_{(b,z,n^*)}$	Scaled integer or Float [w/m2 sr um]	N_{band} x (N_{SCA-1}) x N_n	Y	Temporary file
Line & sample coordinated for sample n in L1T	$x^{L1T}_{(b,z,n)}, y^{L1T}_{(b,z,n)}$	Integer	N_{band} x (N_{SCA-1}) x N_n	Y	Temporary file

Description	Symbol	Units	Level	IAS only	Target
Line & sample coordinated for sample n in L1R from left SCA in boundary zone z (from SCA(z) for OLI or SCA 4-z for TIRS)	$x^{L-L1R}_{(b,z,n)}, y^{L-L1R}_{(b,z,n)}$	Float	N_{band} x (N_{SCA-1}) x N_n	Y	Temporary file
Line & sample coordinated for sample n in L1R from right SCA in boundary zone z (from SCA(z+1) for OLI or from SCA(3-z) for TIRS)	$x^{R-L1R}_{(b,z,n)}, y^{R-L1R}_{(b,z,n)}$	Float	N_{band} x (N_{SCA-1}) x N_n	Y	Temporary file

Total of up to 31 parameters per overlap zone out of which 23 are core parameters needed in LPGS processing

4.4.14.4 Options

Setting the smoothing parameters level for OLI and TIRS (parameters name SMP_OLI and SMP_TIRS). By default they will be 500 but this could be a WO adjustable parameter (similar to TIRS ghosting blur size adjustable parameter) - possible values range is 1-1500.

This data will be stored in the characterization database by default, for standalone or IAS processing more data will be exported from the processing while the core 23 parameters will be provided in a text output summary report along with header information containing scene ID, WRS path row, Instrument ID, sun azimuth, FPA temp, FPA window temps. Report generation should be selectable in IAS work order (default will be off for both LPGS and IAS).

Include in a text report is the rough estimate for SCA overlap statistics derived FPM rel gains and the temporary band average normalized SCA correction factors in each band as well as the normalization factor of mean_SCA_cf[band].

All IAS related outputs that are exported to file should be selectable to be set on or off.

4.4.14.5 Procedure

For each band b in each Earth scene

1. produce the input of $Q^{L}_{(b,z,n^*)}$, $Q^{R}_{(b,z,n^*)}$ arrays for co-registered L1T pixels from right side and left side SCAs in each overlap zone and store the number of pairs found in each overlap zone in $N_{(b,z)}$

for all inputs compute
Ratio(b,z,n*) = $Q^{L}_{(b,z,n^*)} / Q^{R}_{(b,z,n^*)}$

Difference(b,z,n*) = $Q^{L}_{(b,z,n*)}$ - $Q^{R}_{(b,z,n*)}$

Where:

Z is all overlap zones present in the instrument (1-13 for OLI and/or 1-2 for TIRS)

n* is the along track filtered set of co-registered pixels in each overlap

2. compute the following steps (step 3 through 10) for each overlap zone Z in band b

3. compute and store mean and stdev in each overlap zone. (same processing as in Histogram-stat)

r_mean$_{(b,z)}$ = mean(Ratio(b,z,n*)) *d_mean*$_{(b,z)}$ = mean(Difference(b,z,n*))

r_sd$_{(b,z)}$ = stddev(Ratio(b,z,n*)) *d_sd*$_{(b,z)}$ = stddev(Difference(b,z,n*))

$Q^{L}\_mean_{(b,z)}$ = mean($Q^{L}_{(b,z,n*)}$) $Q^{L}\_sd_{(b,z)}$ = stddev($Q^{L}_{(b,z,n*)}$)

------ run steps 4-11 for both ratio of responses and difference of responses -------

4. smooth along track data

First, due to possible miss-registration and oversampling, we need to smooth the input data by a moving average of several points (as set by the parameter SMP_OLI and SMP_TIRS – by default those are 500). That will produce a spatio-temporal average smoothing that progresses in the along track for ratio or differences of data in the processed overlap zone.

The desired output is a low frequency along-track response ratio or response difference information.

For the given band b and overlap zone z compute the smoothed ratio or difference Rs_ratio(b,z,n*)=R ; Rs_delta(b,z,n*)=R are produced as shown in equation 1 when A=ratio(b,z,n*) or A=Difference(b,z,n*) and w=SMP_OLI or w=SMP_TIRS depending on the band.

(Eq. 1)

$$\bar{x}[i] = \frac{1}{2M+1} \sum_{j=-M}^{M} x[i+j]$$

Where: 2M+1=w

for all i where M<= i <= (N(b,z)-M-1)

and X_d=A_d for the reminder of the array. (Note: those get ignored anyway in fit process in step 5.)

and also

Ri= $\overline{X}_1$

Alternatively this can be defined as

(Eq. 2)

$$R_i = \begin{cases} \frac{1}{w} \sum_{j=0}^{w-1} A_{i+j-(w-1)/2}, & i = \frac{(w-1)}{2}, \ldots, N - \frac{(w+1)}{2} \\ A_i, \text{ otherwise} \end{cases}$$

Which means that the smoothed output Ri is defined as Ai when i< (w-1)/2 or when i>(N-(w+1)/2)

And for every i in the range between (w-1)/2 <=i<=(N-(w+1)/2)

Ri is equal to the average of Ai over the width w as illustrated by the single sum of the j index

This average is computed as summation of all data points from A[i-(w-1)/2] to A[i+(w-1)/2] divided by width w

5. compute 3rd order polynominal least squares fit to along track in overlap zone along with fit success flag and Chi-square.

use the smothered data from step 4. in the polynomial fit after removing first and last SMP_OLI/2 or SMP_TIRS/2 points from the smoothed ratio or difference.

Define the following

x = long array for the index to smoothed points in along track data in band b zone z

range is from 1 to (N(b,z)– w)

y=Ri from step 4. Without the points

where Ri=Ai (in other words Y is defined between $Y_1=R_{(w-1)/2}$ and

$Y_{N(b,z)-w}=R_{N(b,z)-(w+1)/2}$)

k=3 polynomial order

so when Y fit is defined by

(Eq. 3)

$$y = a_0 + a_1 x + \ldots + a_k x^k,$$

In matrix form the least square fit parameters (a) are computed with the following matrix operation (using the Vandermonde matrix for X)

Where the i^{th} row of x and y matrices will contain the x and y value for the i^{th} data samples as defined above and they will have k+1 columns for each polynomial power X^T is the transpose matrix and $(\ldots)^{-1}$ is the inverted matrix.

Then the least square fit parameters are found using this matrix operation

(Eq. 4)

$$\mathbf{a} = \left(X^T X\right)^{-1} X^T \mathbf{y}.$$

and Yfit from (Eq. 3) could be rewritten as

(Eq. 5)

$$\mathbf{y} = X\,\mathbf{a}.$$

If the least square coefficients got generated without errors then set success flag =1

If error occurred set success flag=0

Assuming no error occurs compute the following

Chi-square is defined as:

(Eq. 6)

$$\chi^2 = \frac{1}{D_f} \sum_i W_i \left(Y_i - Yfit_i\right)^2$$

Where W is the weighting parameter.

We selected to have no weighting so W =1 for all i

Alternatively W could be set to be equal to the variance defined by StDev^2 within each group of subsets of points (500 points by default as defined by SMP_OLI and SMP_TIRS)

And D_f is the parameter for the degrees of freedom

In our processing we set $D_f = 1$.

In each overlap zone the Chi-square then becomes the sum of the squared difference between the along track smoothed data and the polynomial fit to that data.

Populate output variables a, success flag and Chi-square

Assuming no error occurs compute the reminder steps of 6 through 9

6. compute Pearson correlation (a.k.a R factor coeff) between fit result and the smoothed input data

$$r = \frac{\text{covariance of X and Y}}{(\text{standard deviation of X})(\text{standard deviation of Y})}$$

i.e.

(Eq. 7)

$$r = \frac{\frac{1}{N-1}\sum_{i=0}^{N-1}\left(x_i - \left[\sum_{k=0}^{N-1}\frac{x_k}{N}\right]\right)\left(y_i - \left[\sum_{k=0}^{N-1}\frac{y_k}{N}\right]\right)}{\sqrt{\frac{1}{N-1}\sum_{i=0}^{N-1}\left(x_i - \left[\sum_{k=0}^{N-1}\frac{x_k}{N}\right]\right)^2}\sqrt{\frac{1}{N-1}\sum_{i=0}^{N-1}\left(y_i - \left[\sum_{k=0}^{N-1}\frac{y_k}{N}\right]\right)^2}}$$

When X and Y here in the following steps are defined as follows:

X is the smoothed along track data (Y in step 5)

Y is the polynomial modeled fitted curve produced by the parameters computed in step 5. (Yfit from step 5 Eq 5)

7. find and store maximum value for the absolute value of the residual fit error

(Eq. 8)

$$\text{Max}(\ \text{abs}(X-Y))$$

8. compute and store fit RMSE residual error as Stdev of residual

(Eq. 9)

$$\text{stddev}(X-Y,/\text{double})$$

9. Compute and store number of points in data that have a residual greater than 3-sigma.

(Eq. 10) for OLI bands

n_elements(where(abs(X-Y) gt (OLI_StDev(overlap zone)/sqrt(SMP_OLI)*3)))

or

(Eq. 10) for TIRS bands

n_elements(where(abs(X-Y) gt (TIRS_StDev(overlap zone)/sqrt(SMP_TIRS)*3)))

10. After ratio and difference means have been computed for all zones compute the overlap statistics derived FPM relative gains and the band average normalization factor by following these steps:

For this step we define a temporary variable named the SCA correction factor - SCA_cf

It is a float array in the size of $N_B x N_{SCA}$

For OLI bands

For SCA 1 set the temporary correction factor to 1

SCA_cf[1,band]=1

For SCA 2 set the correction factor to the ratio_mean for zone 1 divided by the corrector factor of SCA1

SCA_cf[2,band]= {$r\_mean_{(b,z)}$ (for z = 1))/ SCA_cf[1,band] }

For SCA 3 set the correction factor to the ratio_mean for zone 2 divided by the corrector factor of SCA2

SCA_cf[3,band]= { $r\_mean_{(b,z)}$ (for z = 2))/ SCA_cf[2,band] }

And so on …. until SCA 14 when the correction factor is set to the ratio_mean for zone 13 divided by the corrector factor of SCA13

SCA_cf[14,band]= { $r\_mean_{(b,z)}$ (for z = 13))/ SCA_cf[13,band] }

Next normalize the correction factors to the band average of all correction factors per band by dividing the correction factors by mean(SCA_cf[1..14,band]). The result is the band average normalized SCA correction factors.

mean_SCA_cf[band]= sum(SCA_cf[sca,band])/14.

Normalized_SCA_cf[sca,band]=SCA_cf/mean_SCA_cf[band]

Finally compute the per SCA overlap statistics derived FPM rel gains by multiplying the each FPM rel gain by its corresponding correction factor.

OSD_FPM rel gain [sca,b]= FPM rel gain [sca,b]* Normalized_SCA_cf[sca,band]

For TIRS bands

For SCA 1 set the temporary correction factor to 1

SCA_cf[1,band]=1

For SCA 2 set the correction factor to the ratio_mean for zone 1 divided by the corrector factor of SCA1

SCA_cf[2,band]= { $r\_mean_{(b,z)}$ (for z = 1))/ SCA_cf[1,band] }

For SCA 3 set the correction factor to the ratio_mean for zone 2 divided by the corrector factor of SCA2

SCA_cf[3,band]= { $r\_mean_{(b,z)}$ (for z = 2))/ SCA_cf[2,band] }

Next normalize the correction factors to the band average of all correction factors per band by dividing the correction factors by mean(SCA_cf[1..3,band]). The result is the band average normalized SCA correction factors.

mean_SCA_cf[band]= sum(SCA_cf[sca,band])/3.

Normalized_SCA_cf[sca,band]=SCA_cf/mean_SCA_cf[band]

Finally compute the per SCA overlap statistics derived FPM rel gains by multiplying the each FPM rel gain by its corresponding correction factor.

OSD_FPM rel gain [sca,b]= FPM rel gain [sca,b]* Normalized_SCA_cf[sca,band]

11. save results to database and generate report file (if option selected in work order)
    Include in report the SCA overlap statistics derived FPM rel gains and the temporary band average normalized SCA correction factors in each band as well as the normalization factor of mean_SCA_cf[band].

4.4.15 Inoperable Detectors Fill

4.4.15.1 Background/Introduction

An inoperable detector is one that provides no change in output DN value when radiance on its input is changed. The data from inoperable detectors appear as very distinct stripes in final image products. This algorithm replaces image data generated by the inoperable detectors with the data from neighboring detectors, in order to enhance visual appearance of OLI and TIRS images. It does not replace data generated by the "out-of-spec" detectors (TIRS may have different way of dealing with "out-of-spec" detectors). The algorithm assumes that the input scene data are nominally spatially aligned and that a list of known inoperable detectors is available within the CPF. The algorithm operates within the normal Level 1R data processing flow.

4.4.15.2 Inputs

Description	Symbol	Units	Level	Source	Type
Scene (L1R) Earth data	L	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA} x N_{det} x N_{frames}		Float
Inoperable detectors			N_{band} x N_{SCA} x N_{det}	CPF	Int

4.4.15.3 Outputs

Description	Symbol	Units	Level	Target	Type
Scene (L1R-corrected)	L	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA} x N_{det} x N_{frame}		Float

4.4.15.4 Procedure

For each inoperable detector, d_i, listed in the CPF

1. If the inoperable detector's immediate neighbors are operable, replace each pixel in the image column generated by the inoperable detector with the mean of two adjacent pixels in the same line, l, of the same SCA:

$$Q(l, d_i) = \frac{Q(l, d_{i-1}) + Q(l, d_{i+1})}{2}$$

- If the inoperable detector is the first (d_1) or last (d_{max}) on an SCA, its data will be replaced with the data from the nearest neighboring operable detector from the same SCA.

$$Q(l, d_1) = Q(l, d_2)$$

$$Q(l, d_{max}) = Q(l, d_{max-1})$$

2. If there are two consecutive inoperable detectors, d_i and d_{i+1}, on the same SCA, then the correction of corresponding image data needs to be accomplished by interpolating data from the available adjacent operable detectors:

$$Q(l, d_i) = \frac{2 \times Q(l, d_{i-1}) + Q(l, d_{i+2})}{3}$$

$$Q(l, d_{i+1}) = \frac{Q(l, d_{i-1}) + 2 \times Q(l, d_{i+2})}{3}$$

For example, if detectors 23 and 24 are inoperable, then their outputs need to be replaced with the result of linear interpolation of data from detectors 22 and 25, for each line of L1R image.

- If the two inoperable detectors are the first and second detector on an SCA, their data will be replaced with the data from the third detector from the same SCA.

$$Q(l, d_1) = Q(l, d_3)$$

$$Q(l, d_2) = Q(l, d_3)$$

- If the two inoperable detectors are the last and second last detectors on an SCA, their data will be replaced with the data from the third last detector from the same SCA.

$$Q(l, d_{\max}) = Q(l, d_{\max-2})$$

$$Q(l, d_{max-1}) = Q(l, d_{max-2})$$

3. If there are more than two consecutive inoperable detectors on an SCA, then the pixel values for the affected detectors need to be filled with zeros.

Note: To replace inoperable detector data, this algorithm uses pixels affected by artifacts, e.g., impulse noise or saturation, and fill pixels (on top and bottom of images that support the nominal image alignment) the same way as regular pixels.

4.4.16 Saturated Pixel Replacement

4.4.16.1 Background/Introduction

The saturated pixels determined in L0R data using the Saturated Pixel Characterization algorithm may change their values during radiometric processing. As a result, they may not be easily identified in final L1R product. To avoid erroneous interpretation of radiometric data in L1R products, it is important to clearly locate originally identified saturated pixels.

This algorithm describes the post-1R correction to replace high-end saturated pixels in image data with the band-maximum radiance (L_{sat_max}) values and low-end saturated pixels with the band-minimum radiance (L_{sat_min}) values. The algorithm operates within the normal Level 1R data processing flow.

4.4.16.2 Inputs

Description	Symbol	Units	Level	Source	Type
Scene (L1R) Earth data	*L*	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA} x N_{det} x N_{frame}		Float
Saturated pixel locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
Low Saturation Radiance Level	L_{sat_min}	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band}	CPF	Float
High Saturation Radiance Level	L_{sat_max}	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band}	CPF	Float

4.4.16.3 Outputs

Description	Symbol	Units	Level	Target	Type
Scene (L1R-corrected)	*L*	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA} x N_{det} x N_{frame}		Float

4.4.16.4 Options

By default, the digitally low and high saturated pixels will be replaced with the corresponding low and high saturation radiance levels. In addition, the following options need to be selectable through work order parameters:

- replace pixels only at the high saturation level or only at the low saturation level
- replace both analog and digitally saturated pixels (with same radiance saturation levels).
- no saturated pixel replacement.

4.4.16.5 Procedure

For each detector (d):

1. Based on the selected options and Labeled Mask record, find each pixel originally identified in Saturated Pixel Characterization algorithm as low and/or high saturated

2. Replace the pixel values with the corresponding Low and High Saturation Radiance Levels, given in CPF

4.4.17 Radiance Rescaling

4.4.17.1 Background/Introduction

The standard L8/9 products are OLI reflectance and TIRS radiance products in 16-bit integer format. The OLI and TIRS data are radiometrically and geometrically processed using floating point operations. For OLI, this algorithm scales and converts the resultant L1G reflectance from floating point format to 16-bit integer format using scaling parameters from the CPF. In addition, the algorithm provides rescaling coefficients for direct conversion from 16-bit integer reflectance to floating point radiance. For TIRS, the algorithm scales and converts radiance values from floating point format to 16-bit integer format. The algorithm assumes no scaling is applied during Geometric Processing.

4.4.17.2 Inputs

Description	Symbol	Units	Level	Source	Type
For OLI:					
Reflectance scene (L1G)	ρ		N_{band} x N_{SCA} x N_{det} x N_{frame}	Geometric Processing	float
Reflectance multiplicative scaling factor	M_ρ	DN^{-1}	N_{band}	CPF	float
Reflectance additive scaling factor	A_ρ		N_{band}	CPF	float
Reflectance to Radiance conversion coefficient	ρ_R	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band}	Reflectance Conversion	float
For TIRS:					
Radiance scene (L1G)	L	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA} x N_{det} x N_{frame}	Geometric Processing	float
Radiance multiplicative scaling factor	M_L	$\frac{W}{m^2 \cdot sr \cdot \mu m} / DN$	N_{band}	CPF	float
Radiance additive scaling factor	A_L	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band}	CPF	float

4.4.17.3 Outputs

Description	Symbol	Units	Level	Target	Type
For OLI:					
Scene L1G	ρ_{int}	DN	N_{band} x N_{SCA} x N_{det} x N_{frame}		Uint (16-bit)
Reflectance multiplicative scaling factor	M_ρ	DN^{-1}	N_{band}	Metadata	float
Reflectance additive scaling factor	A_ρ		N_{band}	Metadata	float
Radiance multiplicative scaling factor	M_L	$\frac{W}{m^2 \cdot sr \cdot \mu m} / DN$	N_{band}	Metadata	float
Radiance additive scaling factor	A_L	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band}	Metadata	float

Description	Symbol	Units	Level	Target	Type
For TIRS:					
Scene (L1G)	L_{int}	*DN*	N_{band} x N_{SCA} x N_{det} x N_{frame}		Uint (16-bit)
Radiance multiplicative scaling factor	M_L	$\frac{W}{m^2 \cdot sr \cdot \mu m} / DN$	N_{band}	Metadata	float
Radiance additive scaling factor	A_L	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band}	Metadata	float

4.4.17.4 Procedure

1. For each band, apply scaling to each image pixel, except to fill data, of geometrically corrected (L1G) floating point OLI reflectance and TIRS radiance images:

    a. For OLI

$$\rho_{scal} = \frac{\rho - A_\rho}{M_\rho}$$

    b. For TIRS

$$L_{scal} = \frac{L - A_L}{M_L}$$

2. Convert scaled OLI reflectance (ρ_{scal}), TIRS radiance (L_{scal}) and fill data pixel values from floating point to 16-bit integer format through rounding to the closest 16-bit integer values

    a. For OLI

$$\rho_{\text{int}} = round\left(\rho_{scal}\right)$$

    b. For TIRS

$$L_{\text{int}} = round\left(L_{scal}\right)$$

3. Under the assumption that 0 will be reserved for fill data, convert all zeros in ρ_{int} and L_{int} images to Q_{calmin} = 1. More generally, convert all pixels with value less than Q_{calmin} to Q_{calmin}.

4. Only for OLI, calculate rescaling coefficients that will be used for direct conversion from 16-bit integer reflectance to floating point radiance

$$M_L = \rho_R \cdot M_\rho$$
$$A_L = \rho_R \cdot A_\rho$$

5. Write the following rescaling parameters to the product metadata

    a. For OLI:
        i. Reflectance multiplicative scaling factor, M_ρ
        ii. Reflectance additive scaling factor, A_ρ
        iii. Radiance multiplicative scaling factor, M_L
        iv. Radiance additive scaling factor, A_L

    b. For TIRS:
        i. Radiance multiplicative scaling factor, M_L
        ii. Radiance additive scaling factor, A_L

4.4.18 Cloud Cover Assessment CCA – CFMask

4.4.18.1 Background/Introduction

Throughout the years of USGS Landsat data processing, many cloud cover algorithms have been applied to the L1 products, each an assumed improvement over the past versions used. Over time, the multiple algorithms and data processing techniques caused inconsistent L1 data products in the Landsat data archive.

During the 2016 Collection 1 processing effort, CFMask was determined to be the single standard CCA algorithm.

CFMask is the C language version of the Function of Mask algorithm. It was originally researched at Boston University and developed further at USGS EROS. CFMask is a multi-pass algorithm that uses decision trees to prospectively label pixels in the scene, then validates or discards those labels according to scene-wide statistics.

CFMask is attractive as a solution to Landsat cloud masking because of its high accuracy and because it includes a cloud shadow detection algorithm. Refinements in CFMask include the ability to operate without thermal data, the ability to use cirrus band data, and variant algorithms that allow it to run on data from several Landsat-like instruments.

The output of CFMask is an intermediate CCA mask which is then used to create the final scene QA mask.

4.4.18.2 Inputs

Description	Units	Level	Source	Type
Reflectance band scene data, as TOA reflectance	Refl. Units	Scene, OLI bands 2-7, 9 or TM/ETM+ bands 1-5 and 7		float

Description	Units	Level	Source	Type
Thermal band scene data, as TOA Brightness Temperature	Celsius	Scene, TIRS Band 1 or TM/ETM+ band 6		float
Scene Elevation image	meters	scene	DEM	long

4.4.18.3 Outputs

Description	Potential Values
Fill	Values (0, 1, 2, etc.) Bits (00, 01, 10, 11)
Water	Values (0, 1, 2, etc.) Bits (00, 01, 10, 11)
Cloud Shadow	Values (0, 1, 2, etc.) Bits (00, 01, 10, 11)
Snow/Ice	Values (0, 1, 2, etc.) Bits (00, 01, 10, 11)
Cloud	Values (0, 1, 2, etc.) Bits (00, 01, 10, 11)
Cloud Confidence	Values (0, 1, 2, etc.) Bits (00, 01, 10, 11)

4.4.18.4 Procedure

CFMask involves several passes. In each pass, the algorithm evaluates each non-fill pixel in the scene.

1. Pre-defined variables
    a. t_buffer = 4.0
    b. cloud_prob_threshold = 22.5

4.4.18.4.1 Pass 1 – Basic tests

For each pixel in the scene: If any satellite other than Landsat 8 or 9, if this pixel is at the TOA saturation value for any band, set that band's pixel value to the TOA maximum value.

Calculate NDVI and NDSI

1. NDVI = (r – nir)/(r + nir) (If r+nir == 0, set NDVI = 0.01)
2. NDSI = (g – swir1)/(g + swir1) (If g+swir1 == 0, set NDSI = 0.01)
3. If any visible band (b, g, r) is saturated, mark pixel as saturated.

Perform Basic Cloud Test

1. If (NDSI < 0.8 and NDVI < 0.8 and swir2 > 0.03), then positive result.
2. If result is positive and not using thermal, then mark pixel as cloud.
3. If result is positive and using thermal, and thermal < 27 C, then mark pixel as cloud.
    a. If pixel is marked as cloudy:
        1. Perform whiteness test.
            a. Calculate Visible Mean
                i. visi_mean = (b + g + r)/3.0

b. Calculate Whiteness
    i. If (visi_mean = 0.0) then whiteness = 100.0
    ii. If any satellite other than Landsat 8 or 9, and any visible band (R, G, B) is saturated, then whiteness = 0.0. Also set a saturation flag (satu_bv) that will be used in the HOT tests.
    iii. Else, whiteness = (abs(b + g + r – visi_mean))/visi_mean
c. If pixel is marked as cloud and whiteness >= 0.7, then mark this pixel as clear.

Perform Haze Optimized Transformation (HOT)

1. If ((b – r/2) <= 0.08) and the pixel is not saturated (satu_bv is 0), then mark this pixel as clear.
2. If (swir1 is not 0.0) and (nir/swir1 <= 0.75), then mark this pixel as clear.
    a. If using the cirrus band, if (cirrus/4.0 > 0.0025) then mark this pixel as cloud.

Perform Basic Snow Test

1. If (NDSI > 0.15 and nir > 0.11and g > 0.1) then result is positive.
2. If result is positive and not using thermal, then mark pixel as snow.
3. If result is positive and using thermal, and thermal < 10 C, then mark pixel as snow.

Perform Basic Water Test

1. If (NDVI < 0.01 and nir < 0.11) or (NDVI < 0.1 and NDVI > 0.0 and nir < 0.05) then mark pixel as water.
2. If pixel is not cloud, increment the Clear Count.
    a. Set pixel's clear_bit.
    b. If pixel is water and not cloud, increment the Clear Water Count.
        i. Set pixel's clear_water_bit.
    c. If pixel is not water and not cloud, increment the Clear Land Count.
        i. Set pixel's clear_land_bit.

Calculate Clear and Water Statistics

1. Calculate clear percentage.
    a. clear_ptm = Clear Count / Total Non-Fill Image Pixels.
2. Calculate clear water percentage.
    a. water_ptm = Clear Water Count / Total Non-Fill Image Pixels.
3. Calculate clear land percentage.
    a. land_ptm = Clear Land Count / Total Non-Fill Image Pixels.
4. If clear_ptm <= 0.1, then assume the entire scene is cloudy or cloud shadowed.
    a. Mark all non-cloudy pixels in scene as cloud shadow.
    b. Mark all cloudy pixels in the scene as high-confidence cloud.
    c. If using thermal, then disable the thermal thresholds:
        i. Set t_templ = -1.0
        ii. Set t_temph = -1.0
5. If land_ptm >= 0.1 then expect clear land.
    a. Set land_bit = clear_land_bit.

b. else set land_bit = clear_bit.
6. If water_ptm >= 0.1 then expect clear water.
   a. Set water_bit = clear_water_bit.
   b. else set water_bit = clear_bit.

4.4.18.4.2 Pass 2 – Calculate Temperature Statistics

For each pixel in the scene: If any satellite other than Landsat 8 or 9, if this pixel is at the TOA saturation value for any band, set that band's pixel value to the TOA maximum value.

If pixel is land and using thermal, calculate land brightness temperature. If pixel is water and using thermal, calculate water brightness temperature.

While calculating brightness temperatures, remember the minimum and maximum land brightness temperatures and the minimum and maximum water brightness temperatures in the scene. Also keep tallies of the number of land pixels and water pixels. If there are no clear land pixels, set minimum and maximum land temperature to zero. If there are no clear water pixels, set minimum and maximum water temperature to zero.

If using thermal, calculate temperature percentiles:
1. t_templ = 17.5% percentile land temperature – t_buffer
2. t_temph = 82.5% percentile land temperature + t_buffer
3. t_wtemp = 82.5% percentile water temperature

4.4.18.4.3 Pass 3 – Calculate Cloud Probability

For each pixel in the scene: If any satellite other than Landsat 8 or 9, if this pixel is at the TOA saturation value for any band, set that band's pixel value to the TOA maximum value.

If pixel is water, calculate brightness probability
1. brightness_prob = swir1/0.11, clipped to between 0.0-1.0.
2. If using thermal, wtemp_prob = (t_wtemp – thermal BT)/4.0.
   a. If wtemp_prob < 0.0, set = 0.0.
   b. brightness_prob = brightness_prob * wtemp_prob
3. If using the cirrus band, then brightness_prob = brightness_prob + cirrus reflectance/0.04.
4. wfinal_prob = 100.0 * brightness_prob

If pixel is land, calculate modified NDVI and modified NDSI
1. NDVI = (r – nir)/(r + nir)
   a. If r+nir == 0, set NDVI = 0.01.
2. NDSI = (g – swir1)/(g + swir1)
   b. If g+swir1 == 0, set NDSI = 0.01.
3. Clip both NDVI and NDSI to positive values.
   a. If less than 0.0, set equal to 0.0.

Calculate whiteness
If any satellite other than Landsat 8 or 9, and any visible band (R, G, B) is saturated, then whiteness = 0.0.

1. Calculate Visible Mean
    a. visi_mean = (b + g + r)/3.0
2. Calculate Whiteness
    a. If (visi_mean = 0.0) then set whiteness = 0.0
    b. Else, whiteness = (abs (b + g + r – visi_mean))/visi_mean

Calculate probabilities.

1. vari_prob = 1.0 - maximum of (max(abs(NDVI), abs(NDSI)), whiteness)
2. If using thermal,
    a. temp_prob = (t_temph – thermal BT)/(t_temph – t_templ). (If temp_prob < 0.0, set = 0.0.)
    b. vari_prob = vari_prob * temp_prob
3. If using cirrus, then vari_prob = vari_prob + cirrus reflectance/0.04.
4. final_prob = 100.0 * vari_prob

Calculate dynamic land cloud threshold

1. Threshold clr_mask = 82.5% percentile value of final_prob pixels that are also marked as land_bit.
2. Add cloud_prob_threshold to clr_mask.

Calculate dynamic water cloud threshold.

1. Threshold wclr_mask = 82.5% percentile value of wfinal_prob pixels that are also marked as water_bit.
2. Add cloud_prob_threshold to wclr_mask.

4.4.18.4.4 Pass 4 – Assign confidence levels

For each pixel in the scene:

1. If using thermal and thermal BT < (t_templ + t_buffer - 35.0), then mark pixel as high confidence cloud and skip the remaining tests.
2. If pixel is water, is marked as cloud, and wfinal_prob > wclr_mask, then mark pixel as high confidence cloud.
3. If pixel is land, is marked as cloud, and final_prob > clr_mask, then mark pixel as high confidence cloud.
4. If pixel is water, is marked as cloud, and wfinal_prob > wclr_mask – 10.0, then mark pixel as medium confidence cloud.
5. If pixel is land, is marked as cloud, and final_prob > clr_mask – 10.0, then mark pixel as medium confidence cloud.
6. In all other cases, mark pixel as low confidence cloud.

4.4.18.5 Pass 5 – Potential Cloud Shadow Mask

Calculate Flood filling Statistics

If any satellite other than Landsat 8 or 9, then for each pixel in the scene, if this pixel is at the TOA saturation value for the NIR or SWIR1 band, set that band's pixel value to the TOA maximum value.

1. Calculate min and max values for both the nir and swir1 bands.
2. Calculate 17.5% percentile of both nir and swir1 bands.
3. Use percentile values to create a flood-filled image for both nir and swir1.

For each pixel in the scene: If any satellite other then Landsat 8 or 9, then for each pixel in the scene, if this pixel is at the TOA saturation value for the NIR or SWIR1 band, set that band's pixel value to the TOA maximum value.

1. Shadow probability = minimum of (new nir band or new swir1 band).
2. If shadow probability > 0.02 (in reflectance units) and the pixel is not marked as water, then mark the pixel as potential shadow. Otherwise, mark it as not shadow.

Cloud Shadow Detection

1. If scene is less than 10% clear, do not run shadow processing; set all non-cloud pixels to shadow.
2. Calculate projection angle of clouds to ground.
3. Map cloud pixels to cloud objects, each containing N pixels.
   d. Cloud objects with N < 9 pixels are discarded and not used for shadow calculation.
4. For each cloud object:
   a. Set thresholds for cloud shadow matching.
      i. If cloud is more than 10% of scene area, t_similar = 0.1 and t_buffer = 0.98. This allows for more lenient matching of large clouds that may have shadows outside the scene borders.
      ii. Otherwise, t_similar = 0.3 and t_buffer = 0.98.
   b. If using thermal:
      i. Calculate the min and max temperature in all pixels of the cloud as temp_obj_min and temp_obj_max.
      ii. Estimate cloud radius: rad = $\sqrt{\frac{N}{2*\pi}}$
      iii. If cloud radius is less than the minimum valid cloud size (rad < num_pix), then set cloud temperature t_obj = temp_obj_min.
      iv. If cloud radius is greater than the minimum valid cloud size, calculate pct_obj = $\frac{(rad-3)^2}{rad^2}$
         1. Calculate t_obj as the pct_obj% percentile of the temperature (between temp_obj_min and temp_obj_max). Example: If pct_obj = 0.5, then t_obj is at the 50% percentile between the min and max temperatures.
      v. Calculate height range.
         1. min_height = (10.0 * (t_templ - t_obj) /9.8)

a. Clip min_height to a minimum of 200.
2. max_height = 10.0 * (t_temph - t_obj)
a. Clip max_height to a maximum of 12,000.
c. If not using thermal, then min_height = 200 and max_height = 12,000.

5. Calculate step size.
a. i_step = (2.0 * pixel_size * tan(sun_elevation))
b. If i_step is less than 2*pixel_size, clip it to 2*pixel_size.
6. Cloud Height Iteration
a. For each cloud base height (base_h) from min_height to max_height, in steps of i_step...
i. If using thermal, calculate cloud heights for each pixel in this cloud object:
1. cloud_height[x] = (10.0*(t_obj – temp_pixel[x])/6.5) + base_h
ii. If not using thermal, cloud_height[x] = base_h.
iii. Find the true position of this cloud object using this trial height and the scene DEM.
iv. For each pixel in this cloud, project the pixel onto the ground
1. i_xy = $\frac{cloud\_height[h]}{pixel\_size * \tan(sun\_elevation)}$
2. x' = x + i_xy * cos(sun azimuth)
3. y' = y + i_xy * sin(sun azimuth)
v. Count up all cloud pixels that are matched with projected ground pixels that are fill, cloud, or preliminary shadow as match_all. Also count the number of pixels projected outside the scene borders (out_all) and the total number of pixels in the cloud (total_all). Add out_all to both match_all and total_all, so as to correct the match weighting for shadows that may be outside the scene.
1. Search for the first maximum with more than t_similar% of the cloud pixels matched with a projected potential shadow pixel on the ground. But allow for a small percentage of variation before settling on a maximum. So:
i. Calculate thresh_match = $\frac{match\_all}{total\_all}$
ii. If thresh_match is higher than the record maxima, set the record to equal the thresh_match and remember the cloud heights at this maxima.
iii. If thresh_match is lower than the record maxima but greater than t_buffer*record, continue with the next trial height. This is to allow small variations around the maxima, in hopes that a greater maxima will be found later.
iv. If thresh_match is lower than the record maxima, lower than t_buffer*record, and the record maxima is greater than t_similar, then the record maxima is considered best and the recorded cloud heights are correct.
v. If the record maxima is over 0.95, then assume it is the best possible match and the recorded cloud heights are correct.

vi. Once the correct heights are found:
    1. Re-calculate the true cloud position using this height and the scene DEM.
    2. Re-project the cloud pixels to the ground by recalculating i_xy, x', and y'.
    3. For every pixel in this cloud object, if its projection onto the ground is marked as potential cloud shadow, mark that pixel as verified cloud shadow.
    4. Escape the height loop and continue with the next cloud object.
vii. If thresh_match is lower than the record maxima and the record maxima is not greater than t_similar, continue with the next trial height. This may result in no verified shadow pixels for this cloud object, if the t_similar threshold is never reached.

Mask Dilation

1. Predefined dilation size: cldpix = 3
2. For every pixel (x,y) in the scene, if it is not fill and is marked as cloud, set the cloud bit for every pixel from (x-cldpix, y-cldpix) to (x+cldpix, y+cldpix).

For every pixel (x,y) in the scene, if it is not fill and is marked as shadow, set the shadow bit for every pixel from (x-cldpix, y-cldpix) to (x+cldpix, y+cldpix).

4.4.19 Pseudo-Invariant Calibration Sites (PICS) Characterization

4.4.19.1 Background/Introduction

This algorithm calculates basic statistics from geographic regions of interest (ROI) extracted from geometrically corrected Landsat products. The ROIs can be defined as polygons with vertices given as latitude/longitude coordinates or by a ROI center latitude/longitude coordinates and a width and height of the region. Although the algorithm can be applied to any site, its primary purpose is to repeatedly characterize uniform and stable radiometric pseudo-invariant calibration sites (PICS), save results to the database, and thus enable an automatic monitoring of temporal stability of OLI and TIRS instruments.

4.4.19.2 Inputs

Description	Symbol	Units	Level	Source	Type
L8/9 product (scaled reflectance / radiance), including the Quality band	Q_{cal}	*DN*	L1GT / L1T	Geometric Processing	Unsigned 16-bit int
Reflectance multiplicative scaling factor	M_ρ	DN^{-1}	N_{band}	CPF	Double
Reflectance additive scaling factor	A_ρ	*unitless*	N_{band}	CPF	Double
Radiance multiplicative scaling factor	M_L	$\frac{W}{m^2 \cdot sr \cdot \mu m} / DN$	N_{band}	CPF	Double
Radiance additive scaling factor	A_L	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band}	CPF	Double

Description	Symbol	Units	Level	Source	Type
Thermal conversion constant 1	K_1	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band}	CPF	Double
Thermal conversion constant 2	K_2	K	N_{band}	CPF	Double
ROI corner or center coordinates (lat/lon)		decimal degrees	$N_{corners}$ x N_{ROI} x 2	Shape file	Double
ROI width and height (if ROI is defined by the center coordinates)		pixels	N_{ROI} x 2	Shape file	Double
Line-Of-Sight model					
Geometric grid					
Digital Elevation Model					

ROIs are defined using scene specific shape files. The shape files are ASCII files containing lists of latitude/ longitude coordinates that represent the vertices of polygons that shape defined ROIs. The regions can also be described by a center latitude/ longitude and a width and height (in pixels), as region 2 in the figure below. Multiple ROIs, separated by blank lines, may exist in a single shape file and the two types of regions can be mixed. Shape file names have the form: wrs2_<path>_<row>.shp), where <path> and <row> specify the 3 digit path and row values which contain the regions defined by the file. The WRS association is required by the IAS for automatically running the PICS algorithm for specified sites, but the algorithm may be run using a shape file with any name, if so desired. There are no naming requirements for the region identifiers, but it is recommended to include user information, such as initials and/or the affiliated institution designation.

```
Filename

region_1_identifier
latitude_1 longitude_1
latitude_2 longitude_2
...
latitude_n longitude_n

region_2_identifier
center_latitude center_longitude
width height

...

region_N_identifier
latitude_1 longitude_1
latitude_2 longitude_2
...
latitude_n longitude_n
```

Figure 4-97. Shape file description

4.4.19.3 Outputs

Description	Symbol	Units	Level	Target	Type
Reflectance minimum	ρ'_{min}		N_{band} x N_{ROI}	Db / Report	Double
Reflectance maximum	ρ'_{max}		N_{band} x N_{ROI}	Db / Report	Double
Reflectance mean	$\bar{\rho}'$		N_{band} x N_{ROI}	Db / Report	Double
Reflectance standard deviation	σ_ρ		N_{band} x N_{ROI}	Db / Report	Double
Radiance minimum	L_{min}	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{ROI}	Db / Report	Double
Radiance maximum	L_{max}	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{ROI}	Db / Report	Double
Radiance mean	$\bar{L}$	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{ROI}	Db / Report	Double
Radiance standard deviation	σ_L	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{ROI}	Db / Report	Double
Brightness temperature minimum	T_{min}	K	N_{ROI}	Db / Report	Double
Brightness temperature maximum	T_{max}	K	N_{ROI}	Db / Report	Double
Brightness temperature mean	$\bar{T}$	K	N_{ROI}	Db / Report	Double
Brightness temperature standard deviation	σ_T	K	N_{ROI}	Db / Report	Double
Coefficient of variation	C_v		N_{band} x N_{ROI}	Report	Double
Sun azimuth	φ	degrees	N_{band} x N_{ROI}	Db / Report	Double
Sun elevation	θ_{SE}	degrees	N_{band} x N_{ROI}	Db / Report	Double
Earth-Sun distance	d	AU	N_{band} x N_{ROI}	Db / Report	Double
Total number of pixels in ROI	N_{Total}	Pixels	N_{band} x N_{ROI}	Db / Report	Int
Valid pixels percentage	N_{Valid}	%	N_{band} x N_{ROI}	Db / Report	Int
Fill pixels percentage	N_{Fill}	%	N_{band} x N_{ROI}	Db / Report	Int
Cloud cover percentage	N_{CC}	%	N_{band} x N_{ROI}	Db / Report	Int
ROI identifier			N_{band} x N_{ROI}	Db / Report	String

In addition to the characterization parameters above, the algorithm generates and outputs thumbnail images of ROIs for each band, in ENVI format.

Note: The reflectance (ρ') statistics are calculated and saved without correction for sun angle.

4.4.19.4 Options

Typically, the characterization data are stored in the characterization database. For stand-alone processing, all these data may be output to a summary report if the report generation option is selected in work order.

4.4.19.5 Procedure

1. For each spectral band in the L8/9 product:
    a. Extract Regions Of Interest (ROIs) defined in the associated shape file
    b. For each ROI:
        i. Generate a georeferenced thumbnail image in ENVI format
2. Using Band 2 (blue band) ROIs:
    a. Find the latitude and longitude of the region centroids
    b. Calculate the Earth-Sun distance, d, and solar elevation, θ_{SE}, and azimuth, φ, angles for the centroids using the available IAS tools
3. For each spectral band, each ROI:
    a. Count the total number of pixels, N_{Total}
    b. Flag and calculate the number of pixels in ROI that fall outside of the band image area (filled with zeros). Calculate the percentage of filled pixels in the ROI, N_{Fill}
    c. Calculate the percentage of valid pixels in the ROI, N_{Valid}. Valid pixels are currently defined as all pixels within the ROI except of filled pixels.
    d. Convert the product DNs to TOA spectral radiance, L_λ

$$L_\lambda = M_L \cdot Q_{cal} + A_L$$

        i. Find the TOA radiance minimum, L_{min}
        ii. Find the TOA radiance maximum, L_{max}
        iii. Calculate the TOA radiance mean, $\overline{L}$
        iv. Calculate the TOA radiance standard deviation, σ_L
4. Convert the product DNs to TOA spectral reflectance (without correction for solar angle), ρ_λ'

$$\rho_\lambda' = M_\rho \cdot Q_{cal} + A_\rho$$

    a. For valid pixels:
        i. Find the TOA reflectance minimum, ρ_{min}
        ii. Find the TOA reflectance maximum, ρ_{max}
        iii. Calculate the TOA reflectance mean, $\overline{\rho}$
        iv. Calculate the TOA reflectance standard deviation, σ_ρ

v. Calculate the coefficient of variation, C_v:

$$C_v = \frac{\bar{\rho}}{\sigma_\rho} = \frac{\bar{L}}{\sigma_L}$$

5. Convert the spectral radiances of thermal bands to brightness temperature, T

$$T = \frac{K_2}{ln\left(\frac{K_1}{L_\lambda} + 1\right)}$$

a. For valid pixels in each ROI:

i. Find the brightness temperature minimum, T_{min}

ii. Find the brightness temperature maximum, T_{max}

iii. Calculate the brightness temperature mean,$\bar{T}$

iv. Calculate the brightness temperature standard deviation, σ_T

v. Calculate the coefficient of variation, C_v:

$$C_v = \frac{\bar{T}}{\sigma_T}$$

6. Based on the results of cloud cover assessment, as reported in the Quality Band, calculate the percentage of pixels in ROI contaminated by clouds, N_{CC}
7. Store the statistics to the database and/or report file

4.5 OLI Radiometry Algorithms

4.5.1 OLI Bias Model Calibration

4.5.1.1 Background/Introduction

The focal plane assembly for the OLI includes 14 SCAs (or FPMs) when the bandpass filters are included). Each SCA contains the active detector elements for all the 9 imaging bands and a blind band, plus video reference pixels (VRPs) at both ends of each row of active detectors. The VRPs consist of all the electronics of a normal active detector minus the detector itself, which is replaced by a capacitor. These VRPs thus are sensitive to most of the same electronic variation as the active pixels, minus photo sensitivity. There are 6 VRPs at each end of an SCA for a total of 12 per SCA per band, with the exception of the Pan band, which has 12 at each end for a total of 24. Although included as a design feature, these VRPs were not indicated for use in the initial algorithm description for the bias model. The plan had been to use the blind band (full HgCdTe detectors, though masked from light), to track the dark response of the HgCdTe detectors. Long dark collects taken during Engineering Design Unit (EDU) as well as Flight Unit testing did not show a good correlation of these blind detectors' response to the active response in the absence of light. However, frame to frame variation as well as variation in the collect to collect mean in the dark response of the active pixels was observed to be tracked well by the VRPs. Since these data indicate

that the frame to frame and scene to scene behavior of the VRPs is related to that of the imaging detectors, the data from them are used to estimate a per frame bias for the imaging detectors.

This algorithm calculates three sets of coefficients to estimate the bias of the VNIR and SWIR imaging detectors. First, a per-band, per-SCA scaling factor is calculated that relates the frame to frame behavior of the VRP data to that of the dark response of the imaging detectors. The coefficient is the slope resulting from a linear regression of the per-band, per-SCA, per-frame averages of the imaging detector data and the per-band, per-SCA, per-frame averages of the VRP data from N_0 shutter collects (excluding long shutter collects) acquired near (in time) to the Earth acquisition being corrected. This coefficient is used to estimate the frame to frame variation of the bias within an image. Second, two coefficients, a scaling factor and a constant are determined which are used to estimate the mean of the dark response within an image. These coefficients are from a linear regression of the per FPM means of VRP data and the per-detector means of the imaging detector data from of N_1 shutter collects (excluding long shutter collects) acquired near (in time) to the Earth acquisition being corrected. Saturated pixels and impulse noise pixels, as well as pixels from dropped frames and inoperable detectors, are rejected from the data (both imaging and VRP) prior to fitting the selected model. This algorithm is anticipated to be run for every Earth acquisition as part of processing to produce a level 1R product. Note that the shutter collects included in the N_0 collects are also included in what is anticipated to be the larger group of N_1 shutter collects, and again also that no long shutter collects will be used in this algorithm.

This algorithm should also be implemented as a stand-alone process to enable bias model parameters to be determined for an arbitrary date/time range. This allows for testing the accuracy of the bias estimates.

4.5.1.2 Inputs

Descriptions	Symbol	Unit	Level	Source	Type
Impulse Noise locations in VRPs in shutter collects†			N_0 x N_{band} x N_{SCA} x N_{VRP} x N_{frame}	Mask	Integer
VRP Operability List†			N_1 x N_{band} x N_{SCA} x N_{VRP}	CPF	Integer
Saturated VRP locations in shutter collects†			N_0 x N_{band} x N_{SCA} x N_{VRP} x N_{frame}	Mask	Integer
Dropped Line locations in VRPs in shutter collects†			N_0 x N_{band} x N_{SCA} x N_{VRP} x N_{frame}	Mask	Integer
Impulse Noise locations in shutter collects†			N_0 x N_{band} x N_{SCA} x N_{det} x N_{frame}	Mask	Integer
Saturated Pixel locations in shutter collects†			N_0 x N_{band} x N_{SCA} x N_{det} x N_{frame}	Mask	Integer
Dropped Line locations in shutter collects†			N_0 x N_{band} x N_{SCA} x N_{det} x N_{frame}	Mask	Integer
Shutter Histogram Statistics for N_1 collects immediately preceding current interval and one collect immediately after VRP data included	*S*	DN	(N_{1+1})x$(N_{band\ +1})$x N_{SCA} x N_{det}	Histogram Statistics	Float

Descriptions	Symbol	Unit	Level	Source	Type
Cross track averages of VRP data from shutter collects	A_{VRP}	DN	N_1 x($N_{bands+1}$)x N_{SCA} x N_{frame}	Histogram Statistics	Float
Shutter Data-VRP included	Q and Q_{VRP}	DN	N_0 x N_{band} x N_{SCA} x N_{det} x N_{frame}	L0R	Integer
Nominal Integration Times (MS and Pan)		Micro-seconds	2	CPF	Integer
Collection Integration Times (MS and Pan)		Milli-seconds	2	L0R	Float

*for N pre-acquisition shutter acquisitions; includes VRP data as well. N_{band+1} due to the Pan band. N_{band} does not include the blind band

4.5.1.3 Outputs

Description	Symbol	Unit	Level	Target	Type
Pre-acquisition shutter average (S_A)*	S_A*	DN	(N_{band+1}) x N_{SCA} x N_{det}	BPF	Float
Post-acquisition shutter average (S_B)*	S_B*	DN	(N_{band+1}) x N_{SCA} x N_{det}	BPF	Float
Per-SCA cross-track VRP averages	A_{VRP}	DN	(N_{band+1}) x N_{SCA} x N_{frame}	DB	Float
Bias Model Parameter (a_0)*	a_0	N/A	(N_{band+1}) x N_{SCA}	BPF	Float
Bias Model Parameter (a_1)*	a_1	N/A	(N_{band+1}) x N_{SCA} x N_{det}	BPF	Float
Bias Model Parameter (C_1)*	C_1	N/A	(N_{band+1}) x N_{SCA} x N_{det}	BPF	Float
R-squared values from calculating a_0 (R_0^2)*	R_0^2	N/A	(N_{band+1}) x N_{SCA}	Characterization DB	Float
R-squared values from calculating a_1 and C_1 (R_1^2)*	R_1^2	N/A	(N_{band+1}) x N_{SCA}	Characterization DB	Float

*N_{band+1} accounts for the separation of odd and even lines in the Pan band
N_{band} does not include the blind band

4.5.1.4 Options

- Number of pre-acquisition shutter collects (N_0) and shutter averages (N_1) used for bias model parameter determination (default is N_0=1 and N_1=40)
- Output BPF to file (default off)
- Start and stop date/time of desired bias model parameters (T0 and T1)
    - Normally this is the date/time of the scene being processed.
- The standalone version of this code should have the following options. Two, and only 2 of the 3 options must be chosen.
    - Start date for scenes to be included in N_1
    - End date for scenes to be included in N_1
    - Number of scenes to be included in N_1

  The newest scene within the selected range of scenes will be used for N_0.

4.5.1.5 Procedure

This algorithm can be divided into three sections. In the first section, statistics needed for calculations done later in the algorithm are calculated or pulled from histogram statistics and stored to the database. In the second section, the coefficient a_0, which is used for per frame correction is calculated. In the third section, the coefficients used to estimate the per detector bias mean, a_1 and C_1 are calculated.

Section 1

For all bands

1. Compare the integration time of the data with nominal integration time. Proceed if they are the same.
2. Retrieve the pre-and post-acquisition shutter histogram statistics collected prior to the desired start date/time (T0) and after the desired stop date/time (T1). These statistics are included in *S*, but are now known as S_A and S_B.
3. Send S_A and S_B to the BPF database. This is all that is done with S_B in this algorithm, although S_A is used in later steps.
4. Retrieve N_0 shutter collects (*Q* and Q_{VRP}) prior to the desired start date/time (T0). The first shutter collect before the desired start time will be the pre-acquisition shutter collect. Note that only nominal integration times are used as shutter collects taken during Integration Time Sweeps (ITS) will vary. Also note that anomalous pixels are excluded from ALL calculations in this algorithm. Pixels from inoperable detectors and VRPs are considered anomalous pixels.
5. For each band (except for the pan band) and for each SCA, and treating the data from all selected collects as a single dataset
    a. Find the cross-track average of all of the VRP detectors.
    $$A_{VRP}(b,s,f) = \frac{1}{N_d}\sum_{d=1}^{N_d} Q_{VRP}(b,s,d,f) \quad (1)$$
    where *f* is line, Q_{VRP} is the VRP detector response for the specific band, N_d is the number of operable VRP detectors in a band, and A_{VRP} is the VRP average for frame *f*.
    b. Find the cross-track average of all of the imaging detectors.
    $$A(b,s,f) = \frac{1}{N_d}\sum_{d=1}^{N_d} Q(b,s,d,f) \quad (2)$$
    where *Q* is the imaging detector response for a specific band, N_d here is the number of operable imaging detectors in the specific band, and *A* is the imaging detector average for frame *f.*
6. For the pan band, calculate cross track averages in the same way as in (1) and (2), only treat odd and even frames separately.

$$A_{VRP,even}(b,s,f) = \frac{1}{N_d}\sum_{d=1}^{N_d} Q_{VRP}(b,s,d,f), f = 2,4,6,... \quad (3)$$

$$A_{even}(b,s,f) = \frac{1}{N_d}\sum_{d=1}^{N_d} Q(b,s,d,f), f = 2,4,6,... \quad (4)$$

$$A_{VRP,odd}(b,s,f) = \frac{1}{N_d}\sum_{d=1}^{N_d} Q_{VRP}(b,s,d,f), f = 1,3,5,... \quad (5)$$

$$A_{odd}(b,s,f) = \frac{1}{N_d}\sum_{d=1}^{N_d} Q(b,s,d,f), f = 1,3,5,... \quad (6)$$

7. Write the per-SCA VRP cross-track averages to the database.
8. Retrieve N_1 collects of per detector means (S) and per frame VRP averages (A_{VRP}) from histogram statistics from before the desired start date/time. This includes S_A. Note that all N_1 collects should have been collected at nominal integration time.
9. For each band (except for the Pan band) and each of the N_1 collects, find the per-SCA mean of the VRP data.

$$\overline{A_{VRP}}(b,s,c) = \frac{1}{N_f}\sum_{f=1}^{N_f} A_{VRP}(c,b,s,f) \quad (7)$$

   where A_{VRP} here is the per detector VRP average from histogram statistics, N_f is the number of frames, and c is collect.

10. Calculate the per-SCA VRP means for the pan band in the same way as (7), only treat the averages from odd and even frames separately.

$$\overline{A_{VRP,even}}(b,s,c) = \frac{1}{N_f}\sum_{f=1}^{N_f} A_{VRP,even}(c,b,s,f), f = 2,4,6,... \quad (8)$$

$$\overline{A_{VRP,odd}}(b,s,c) = \frac{1}{N_f}\sum_{f=1}^{N_f} A_{VRP,odd}(c,b,s,f), f = 1,3,5,... \quad (9)$$

Section 2

1. For each band (except for the pan band) and for each SCA, and treating the data from all selected collects as a single dataset
   a. Using a least squares fit, find the coefficient a_0 that relates A_{VRP} to A.

$$A(b,s,f) \approx a_0(b,s)A_{VRP}(b,s,f) \quad (10)$$

   An example of this is plotted in Figure 4-98. The offset determined by the linear regression is ignored, except in calculating the R-squared values.

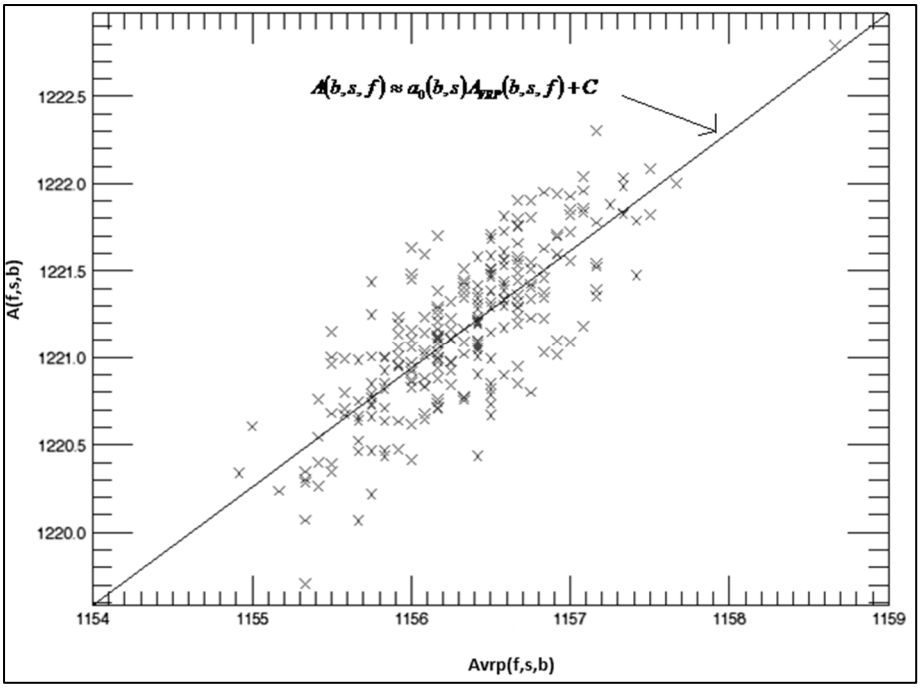

Figure 4-98. VRP Cross Track Average Plotted Against the Imaging Detector Cross Track Average for One SCA of One Band

b. Calculate the R-squared values of the fit, R_0^2

$$A_{estimate}(b,s,f) = a_0(b,s)A_{VRP}(b,s,f) + C \quad (11)$$

$$SS_{err}(b,s) = \sum_{f=1}^{N_f} \left(A_{estimate}(b,s,f) - A(b,s,f)\right)^2 \quad (12)$$

$$\overline{A}(b,s) = \frac{1}{N_f}\sum_{f=1}^{N_f} A(b,s,f) \quad (13)$$

$$SS_{tot}(b,s) = \sum_{f=1}^{N_f} \left(A(b,s,f) - \overline{A}(b,s)\right)^2 \quad (14)$$

$$R_0{}^2(b,s) = 1 - \frac{SS_{err}(b,s)}{SS_{tot}(b,s)} \quad (15)$$

where $A_{estimate}$ is the estimate of the of the A, N_f is the number of frames, and C is the offset calculated during the linear regression. The value will no longer be used after these calculations.

2. Calculate these values for the pan band in the same way as in (10)-(15), only first separate the even frames from the odd frames and calculate two values for both a_0 and R_0^2.

a. $$A_{even}(s,f) \approx a_{0,even}(s) A_{VRP,even}(s,f), f = 2,4,6,... \quad (16)$$

$$A_{odd}(s,f) \approx a_{0,odd}(s) A_{VRP,odd}(s,f), f = 2,4,6,... \quad (17)$$

b. $$A_{estimate,even}(s,f) = a_{0,even}(s) A_{VRP,even}(s,f) + C_{lr,even}, f = 2,4,6,... \quad (18)$$

$$SS_{err,even}(b,s) = \sum_{f=1}^{N_f} \left(A_{estimate,even}(b,s,f) - A_{even}(b,s,f) \right)^2, f = 2,4,6,... \quad (19)$$

$$\overline{A}_{even}(b,s) = \frac{1}{N_f} \sum_{f=1}^{N_f} A_{even}(b,s,f), f = 2,4,6,... \quad (20)$$

$$SS_{tot,even}(b,s) = \sum_{f=1}^{N_f} \left(A_{even}(b,s,f) - \overline{A}_{even}(b,s) \right)^2, f = 2,4,6,... \quad (21)$$

$$R_0{}^2{}_{even}(s) = 1 - \frac{SS_{err,even}(b,s)}{SS_{tot,even}(b,s)} \quad (22)$$

$$A_{estimate,odd}(s,f) = a_{0,odd}(s) A_{VRP,odd}(s,f) + C_{lr,odd}, f = 1,3,5,... \quad (23)$$

$$SS_{err,odd}(b,s) = \sum_{f=1}^{N_f} \left(A_{estimate,odd}(b,s,f) - A_{odd}(b,s,f) \right)^2, f = 1,3,5,... \quad (24)$$

$$\overline{A}_{odd}(b,s) = \frac{1}{N_f} \sum_{f=1}^{N_f} A_{odd}(b,s,f), f = 1,3,5,... \quad (25)$$

$$SS_{tot,odd}(b,s) = \sum_{f=1}^{N_f} \left(A_{odd}(b,s,f) - \overline{A}_{odd}(b,s) \right)^2, f = 1,3,5,... \quad (26)$$

$$R_0{}^2{}_{odd}(s) = 1 - \frac{SS_{err,odd}(b,s)}{SS_{tot,odd}(b,s)} \quad (27)$$

Section 3

1. For each band (except for the Pan band), each SCA and each detector
   a. Using a least squares fit calculate a coefficient a_1 and constant C_1 that relates the means over N_1 collects to the per-SCA VRP means from the same SCA as that detector over the same N_1 collects.

$$S(b,s,d,c) \approx a_1(b,s,d) \overline{A_{VRP}}(b,s,c) + C_1(b,s,d) \quad (28)$$

   b. Calculate the R-squared values from the fit, $R_1{}^2$

$$S_{estimate}(b,s,d,c) = a_1(b,s,d) \overline{A_{VRP}}(b,s,c) + C_1(b,s,d) \quad (29)$$

$$SS_{err}(b,s,d)=\sum_{c=1}^{N_c}\left(S_{estimate}(b,s,d,c)-S(b,s,d,c)\right)^2 \tag{30}$$

$$\overline{S}(b,s,d)=\frac{1}{N_c}\sum_{c=1}^{N_c}S(b,s,d,c) \tag{31}$$

$$SS_{tot}(b,s,d)=\sum_{c=1}^{N_c}\left(S(b,s,d,c)-\overline{S}(b,s,d)\right)^2 \tag{32}$$

$$R_1{}^2(b,s,d)=1-\frac{SS_{err}(b,s,d)}{SS_{tot}(b,s,d)} \tag{33}$$

2. Calculate these values for the pan band in the same way as in (28)-(33) only first separate the even frame means from the odd frame means and calculate two values for each a_1, C_1, and $R_1{}^2$.

   a. $$S_{even}(b,s,d,c)\approx a_{1,even}(b,s,d)\overline{A_{VRP,even}}(b,s,c)+C_{,even1}(b,s,d) \tag{34}$$

   $$S_{odd}(b,s,d,c)\approx a_{1,odd}(b,s,d)\overline{A_{VRP,odd}}(b,s,c)+C_{1,odd}(b,s,d) \tag{35}$$

   b. $$S_{estimate,even}(b,s,d,c)=a_{1,even}(b,s,d)\overline{A_{VRP,even}}(b,s,c)+C_{1,even}(b,s,d) \tag{36}$$

   $$SS_{err,even}(b,s,d)=\sum_{c=1}^{N_c}\left(S_{estimate,even}(b,s,d,c)-S_{even}(b,s,d,c)\right)^2 \tag{37}$$

   $$\overline{S}_{even}(b,s,d)=\frac{1}{N_c}\sum_{c=1}^{N_c}S_{even}(b,s,d,c) \tag{38}$$

   $$SS_{tot,even}(b,s,d)=\sum_{c=1}^{N_c}\left(S_{even}(b,s,d,c)-\overline{S}_{even}(b,s,d)\right)^2 \tag{39}$$

   $$R_{1,even}{}^2(b,s,d)=1-\frac{SS_{err,even}(b,s,d)}{SS_{tot,even}(b,s,d)} \tag{40}$$

   $$S_{estimate,odd}(b,s,d,c)=a_{1,odd}(b,s,d)\overline{A_{VRP,odd}}(b,s,c)+C_{1,odd}(b,s,d) \tag{41}$$

   $$SS_{err,odd}(b,s,d)=\sum_{c=1}^{N_c}\left(S_{estimate,odd}(b,s,d,c)-S_{odd}(b,s,d,c)\right)^2 \tag{42}$$

   $$\overline{S}_{odd}(b,s,d)=\frac{1}{N_c}\sum_{c=1}^{N_c}S_{odd}(b,s,d,c) \tag{43}$$

   $$SS_{tot,odd}(b,s,d)=\sum_{c=1}^{N_c}\left(S_{odd}(b,s,d,c)-\overline{S}_{odd}(b,s,d)\right)^2 \tag{44}$$

   $$R_{1,odd}{}^2(b,s,d)=1-\frac{SS_{err,odd}(b,s,d)}{SS_{tot,odd}(b,s,d)} \tag{45}$$

3. Write a_0, $a_{0,even}$, $a_{0,odd}$, a_1, $a_{1,even}$, $a_{1,odd}$, $C_{1,even}$, and $C_{1,odd}$, to the BPF. Write R_0^2, $R_0^2{}_{,even}$, $R_0^2{}_{,odd}$, R_1^2, $R_1^2{}_{,even}$, and $R_1^2{}_{,odd}$ to the characterization database.
4. If selected, write the BPF to a file

   Note that for any pixel marked as inoperable, dropped, or having been affected by impulse noise or saturation will be excluded from all calculations.

4.5.2 OLI Bias Determination

4.5.2.1 Background/Introduction

Removing the detector's bias from each detector's data is a necessary first step in the conversion from the raw detector signal to radiance (and reflectance) as part of product generation. The bias determination algorithm estimates the bias to remove from each pixel. There are several ways in which the bias can be estimated. One way uses a per-detector bias estimated by averaging shutter data on a per detector basis. Another way uses a pair of coefficients a_1 and C_1 to relate the average per-SCA VRP response to the average per-imaging detector dark response. These coefficients are then applied to the per-SCA VRP response to estimate the average per-detector dark response. Frame to frame variation in the bias can also be estimated by using the VRP data and a coefficient a_0 that relates the per-frame behavior of the VRPs to the per frame behavior of the detector dark response within a scene.

The selection of bias model parameters used to derive the bias to be applied is specified via parameters/flags set in the processing work order. These flags are defined as options below.

4.5.2.2 Inputs

Description	Symbol	Level	Source	Type
VRP cross track averages corresponding to the scene	A_{VRP}	N_{band} x N_{SCA} x N_{det} x N_{frame}	Calculated from image data	Float
Dropped frames in VRP data		N_{band} x N_{SCA} x N_{det} x N_{frame}	Mask	Integer
Impulse noise in VRP data		N_{band} x N_{SCA} x N_{det} x N_{frame}	Mask	Integer
Saturated pixels in VRP data		N_{band} x N_{SCA} x N_{det} x N_{frame}	Mask	Integer
VRP operability list		N_{band} x N_{SCA} x N_{det} x N_{frame}	CPF	Integer
Pre-acquisition shutter average	S_a	N_{band} x N_{SCA} x N_{det}	BPF	Float
Post-acquisition shutter average	S_b	N_{band} x N_{SCA} x N_{det}	BPF	Float
Bias model parameter	a_0	N_{band} x N_{SCA}	BPF	Float
Bias model parameter	a_1	N_{band} x N_{SCA} x N_{det}	BPF	Float
Bias model parameter	C_1	N_{band} x N_{SCA} x N_{det}	BPF	Float
CPF Bias	b_{CPF}	N_{band} x N_{SCA} x N_{det}	CPF	Float

4.5.2.3 Outputs

Description	Level	Target	Type
Bias	N_{band} x N_{SCA} x N_{det} x N_{frame}	Bias Removal or file	Float/Boolean

4.5.2.4 Options

- Output bias values to file (default off)
- bias selection
    - Per-detector bias (no estimate of per frame variation included)
        1. Pre-acquisition shutter average (S_a)
        2. Post-acquisition shutter average (S_b)
        3. Average of pre- and post-acquisition shutter averages (S_{ab})
        4. CPF bias (b_{CPF})
        5. Estimate using a_1, C_1, and the per-SCA VRP averages
    - Per-frame bias (estimate of per frame variation included)
        - Source of a_0, a_1, and C_1
            1. Current BPF
            2. Selected date/time for alternate BPF
        - Five options for calculating C
            1. Pre-acquisition shutter average (S_a)
            2. Post-acquisition shutter average (S_b)
            3. Average of pre- and post-acquisition shutter averages (S_{ab}, default for the per frame bias)
            4. CPF bias (b_{CPF})
            5. Estimate using a_1, C_1, and the per-SCA VRP averages

4.5.2.5 Procedure

If a per-detector bias is selected where the option number is included in 1-4, then retrieve the selected bias values from the BPF. If option 3 has been selected, then calculate S_{AB}, the average of S_A and S_B.

$$S_{AB}(b,s,d) = \frac{1}{2}\left(S_A(b,s,d) + S_B(b,s,d)\right) \quad (1)$$

where d is detector, s is SCA, and b is band.

If a per-detector bias using option 5 is selected, then retrieve a_1 and C_1 from the BPF. Calculate the per-detector bias estimate with these parameters.

For each band excluding the pan band

- For each SCA
    - Find the cross-track average of all of the VRP detectors.

$$A_{VRP}(b,s,f) = \frac{1}{N_d}\sum_{d=1}^{N_d} Q_{VRP}(b,s,d,f) \quad (2)$$

where f is line, Q_{VRP} is the VRP detector response for the specific band, N_d is the number of VRP detectors in a band, and A_{VRP} is the

VRP average for frame *f*. Note that any anomalous pixels will be excluded from calculations.

- Calculate the per-SCA average of the VRP data across all frames.

$$\overline{A_{VRP}}(b,s) = \frac{1}{N}\sum_{f=1}^{N} A_{VRP}(b,s,f) \quad (3)$$

- For each detector
  - Calculate the per-detector estimate of the bias mean.

$$E_{avg}(b,s,d) = a_1(b,s,d)\overline{A_{VRP}}(b,s) + C_1(b,s,d) \quad (4)$$

Where E_{avg} is the estimate of the bias average.

- For the pan band, follow the procedure for the other bands, only treat the odd and even frames separately. Note again that all anomalous pixels are excluded from calculations.

$$A_{VRP,even}(b,s,f) = \frac{1}{N_d}\sum_{d=1}^{N_d} Q_{VRP}(b,s,d,f), f = 2,4,6,... \quad (5)$$

$$A_{VRP,odd}(b,s,f) = \frac{1}{N_d}\sum_{d=1}^{N_d} Q_{VRP}(b,s,d,f), f = 1,3,5,... \quad (6)$$

$$\overline{A_{VRP,even}}(b,s) = \frac{1}{N}\sum_{f=1}^{N} A_{VRP,even}(b,s,f), f = 2,4,6,... \quad (7)$$

$$\overline{A_{VRP,odd}}(b,s) = \frac{1}{N}\sum_{f=1}^{N} A_{VRP,odd}(b,s,f), f = 1,3,5,... \quad (8)$$

$$E_{avg,even}(b,s,d) = a_{1,even}(b,s,d)\overline{A_{VRP,even}}(b,s) + C_{1,even}(b,s,d) \quad (9)$$

$$E_{avg,odd}(b,s,d) = a_{1,odd}(b,s,d)\overline{A_{VRP,odd}}(b,s) + C_{1,odd}(b,s,d) \quad (10)$$

- Write the per-SCA VRP cross track averages to the database.

The result of any of these options should be an array containing a single value for every detector of every band, with the exception being the Pan band where for every detector there is an odd frame estimate and an even frame estimate. Expand this array such that there is one bias value for every frame of the image to be corrected, except for in the case of the Pan band where the even frame estimate should be expanded to only the number of even frames, and the odd frame estimates expanded to only then number of odd frames. The result will be a single value for every pixel in the image to be corrected where every frame belonging to the same detector is identical.

Note: 1.5 DN should have been added to non-barrel-shifted data during convert to float. This is to account for the average error due subtracting the lower 12 of 14-bit data from the upper 12 of 14-bit data.

If the selected option is to have the bias include an estimate of the per-frame variation, calculate the per-frame bias based on bias model parameters from the specified BPF.

- For each band excluding the pan band
  - For each SCA
    - Retrieve the bias model parameters (a_0) from the specified BPF.
    - Calculate the per-SCA average of the VRP data across all frames.

      $$\overline{A_{VRP}}(b,s) = \frac{1}{N}\sum_{f=1}^{N} A_{VRP}(b,s,f) \quad (11)$$

      where $\overline{A_{VRP}}$ is the VRP data average and *N* is the number of frames.

  - For each SCA, detector, and frame
    - If one of the options 1-4 for calculating *C* is selected
      - Retrieve the values of S, where S is per detector bias values from the source selected from the previous list.
      - Calculate the constant value *C*.

        $$C(b,s,d) = S(b,s,d) - a_o(b,s)\overline{A_{VRP}}(b,s) \quad (12)$$

        where *S* is the selected per detector bias. Note that no pixels marked as bad or anomalous are included in this calculation.
    - Otherwise, if option 5 is chosen, then *C* is calculated in exactly the same way as in (12) except S is $E_{avg,}$ as written in (13).

      $$C(b,s,d) = E_{avg}(b,s,d) - a_o(b,s)\overline{A_{VRP}}(b,s) \quad (13)$$

    - Calculate the per frame bias estimate.

      $$bias(b,s,d,f) = a_0(b,s)A_{VRP}(b,s,f) + C(b,s,d) \quad (14)$$

      where *bias* is the bias estimate.
- For the pan band, follow the procedure for the other bands, only treat the odd and the even frames separately.

$$\overline{A_{VRP,even}}(b,s) = \frac{1}{N_{even}}\sum_{f=1}^{N_{even}} A_{VRP,even}(b,s,f), f = 2,4,6,... \quad (15)$$

$$\overline{A_{VRP,odd}}(b,s) = \frac{1}{N_{odd}}\sum_{f=1}^{N_{odd}} A_{VRP,odd}(b,s,f), f = 1,3,5,... \quad (16)$$

$$C_{even}(b,s,d) = S_{even}(b,s,d) - a_{o,even}(b,s)\overline{A_{VRP,even}}(b,s) \quad (17)$$

$$C_{odd}(b,s,d) = S_{odd}(b,s,d) - a_{o,odd}(b,s)\overline{A_{VRP,odd}}(b,s) \quad (18)$$

$$bias_{even}(b,s,d,f) = a_{0,even}(b,s)A_{VRP,even}(b,s,f) + C_{even}(b,s,d) \quad (19)$$

$$bias_{odd}(b,s,d,f) = a_{0,odd}(b,s)A_{VRP,odd}(b,s,f) + C_{odd}(b,s,d) \quad (20)$$

Note that dropped frames will be excluded from calculations and will be set to zero in *bias*.

Send the bias to Bias Removal, and if the option is selected, to file.

4.5.3 OLI Bias Removal

4.5.3.1 Background/Introduction

Conversion to radiance (L1R) occurs in 3 steps: bias removal; response linearization; and gain (absolute and relative) application. This algorithm addresses the first step in generating the L1R radiance product, removing bias. Applying gain and linearizing the detector response are addressed in separate algorithms. Bias removal is accomplished by subtracting a value (in DN) from each pixel of the input image. This value varies by detector for all bands, and also by frame.

Options to select between a CPF bias and biases derived from shutter data acquired near the collect are available for special processing. An option for bias temperature sensitivity correction (described in a separate algorithm) is applied within this algorithm.

4.5.3.2 Inputs

Description	Level	Source	Type
Scene (L0R)	$N_{band,}$ x N_{SCA} x N_{det} x N_{frame}	L0R	Float
Per line correction	N_{band} x N_{SCA} x N_{det} x N_{frame}	Bias Determination	Float
Per detector correction*	(N_{band+1}) x N_{SCA} x N_{det}	CPF	Float
Temperature Correction Factor (CF_T – Unitless)	$N_{band,}$ x N_{SCA} x N_{det}	Temperature Sensitivity Correction	Float

*(N_{band+1}) accounts for the pan band being separated into odd and even detectors

4.5.3.3 Outputs

Description	Level	Target	Type
Bias Corrected Scene	N_{band} x N_{SCA} x N_{det} x N_{frame}	Response Linearization	Float
Choice of Bias	1	L1R Metadata	String
Temperature Sensitivity Correction Flag	1	L1R Metadata	Integer

4.5.3.4 Options

- Apply temperature sensitivity correction (default off)
- Choice of bias
  - From CPF
  - From bias determination algorithm (default)

4.5.3.5 Procedure

For each band, SCA, detector and line

1. If temperature sensitivity correction is selected, multiply the temperature correction factor CF_T by the corresponding bias.

$$bias\ '= CF_T(b,s,d)\cdot bias\ (b,s,d,f) \quad (1)$$

where *b* is band, *s* is SCA, *d* is detector, and *f* is line, and if the bias is to come from the CPF, then *f* in (1) is one.

2. Subtract the per-line or per-detector bias from the corresponding input scene pixel. If temperature sensitivity correction is selected, use (2). Otherwise, use (3).

$$Q'(b,s,d,f) = Q(b,s,d,f) - b'(b,s,d,f) \quad (2)$$

$$Q'(b,s,d,f) = Q(b,s,d,f) - b(b,s,d,f) \quad (3)$$

where *Q* is the input scene data *Q'* is the output bias corrected scene data, and if the bias is to come from the CPF, then *f* in *b* and *b'* in (2) and (3) is one.

4.5.4 OLI Characterize Radiometric Stability (16-day)

4.5.4.1 Background/Introduction

Radiometric stability of an instrument is fundamental to low uncertainty in the radiometric calibration of data products generated from its data. OLI has requirements on its band average stability over several time intervals, specifically 60 second (about 2 scenes), up to 16-day (one repeat cycle) and 16-days up to 5 years (mission lifetime). This algorithm specifically addresses the 16 day radiometric stability.

The key performance metric is the variation in the band average response of the instrument to a constant radiance (greater than or equal to $L_{typical}$) over any period of time up to 16 days. Of the measurements made over these periods, 95% need to be within 1.2% of the average value for all bands but the cirrus band.

This algorithm characterizes the stability of the OLI bands radiometric response using the on-board calibration devices. In particular, the working Stim lamps will be used every day and the working solar diffuser will be used nominally every 8 days.

This algorithm will run on bias-corrected and linearized digital numbers from the Lamp Response Characterization and/or Histogram Statistics Characterization database tables; they do not require separate analysis of image data. After the initial run to generate the traveling average for the history of the mission, the algorithm will run regularly (daily?) to keep the traveling average table populated.

4.5.4.2 Dependencies

Histogram Statistics Characterization
Lamp Characterization
Diffuser Characterization

4.5.4.3 Inputs

Description	Symbol	Units	Level	Source	Type
Lamp Acquisition info: date, time, ID, valid flag				DB	
Level-1 means (bias-corrected, linearized)	$Q_{t,i}$	DN	N_{band} x N_{SCA} x N_{det}	DB (lamp response table)	Float

Description	Symbol	Units	Level	Source	Type
Diffuser acquisition info: date, time, ID, valid flag				DB	
Earth-sun distance	d_t	au		JPL model	float
Level-1 means (bias-corrected, linearized)	$Q_{t,i}$	DN	N_{band} x N_{SCA} x N_{det}	DB (HIST STATS table)	Float
Inoperable detectors, out-of-spec detectors				CPF	integer
Moving window size (multiple window sizes possible, different sizes for lamp and diffuser)	w	Days	1	default=16	integer

4.5.4.4 Outputs

Descriptions	Symbol	Units	Level	Source	Type
calibrator type (LAMP/DIFFUSER)				STABILITY database table	
Time interval grid	tt	days	1	STABILITY database table	int
Window size	w	Days	1	STABILITY database table	int
valid flag	valid Y/N	[]	1	STABILITY database table	int
Number of samples within each traveling average window	n	count	N_{band}	STABILITY database table	int
Traveling average means	$\bar{x}$	DN	N_{band}	STABILITY database table	Float
traveling average stdevs	s_i	DN	N_{band}	STABILITY database table	float
Traveling average metric: Coefficient of Variation	CV	[]	N_{band}	STABILITY database table	float
Traveling average metric: Coefficient of Variation uncertainty	unc_{CV}	[]	N_{band}	STABILITY database table	float
Traveling average metric: Chi^2	Chi^2	[]	N_{band}	STABILITY database table	float
Traveling average metric: Chi^2 uncertainty	unc_{Chi}	[]	N_{band}	STABILITY database table	float

4.5.4.5 Options

Report data to ASCII report files as well as IDL save file and plots.

4.5.4.6 Procedure

For each appropriate collect, for each band:

1. For each appropriate collect, for each band:
    a. Extract Lamp Characterization and Diffuser Characterization database table data for each detector for each acquisition; $Q_{t,i}$
        i. For the lamp, only extract the working lamp data, since the other bulb pairs won't be used often enough to provide meaningful trends.
        ii. For the panel, use only the working panel data
        iii. Calculate time since launch from acquisition dates: *t*
            1. t = acquisition_date – launch_date
        iv. Only consider work orders with valid flag = Y
    b. Calculate band-average means from per-detector means:
        i. $Q_{t,BA} = average(Q_{t,i})$
        ii. The lifetime array will be referred to as Q_{BA}.
        iii. Do not include inoperable or out-of-spec detectors in the average
    c. For diffuser data, correct the band-average signal for the earth-sun distance
$$Q_{t,BA} = Q_{t,BA} * d_t^2$$
2. For the mission lifetime data:
    a. Generate an integer array of dates to have a regular grid over which to compute the traveling average
        i. tt = indgen(max(t)-min(t)-w) + w/2 + min(t)
        ii. grid starts w/2 days into the dataset, so that all averages include data for the whole 16 day window
    b. For each calibrator for the default window (w =16 days):
        i. Calculate the traveling average for the band average signal:
$$\bar{x}_{tt} = traveling\_average(Q_{BA}, tt, w)$$
            1. Do not calculate the average and stdev for windows where there are less than 3 samples.
        ii. Calculate stdev for each of the traveling average windows stdev:
            1. $\sigma_{tt} = stdev(Q_{BA}, tt, w)$
        iii. Record the number of acquisition samples within each traveling average window:
$$n_{tt} = count(Q_{BA}, tt, w)$$
        iv. Coefficient of Variation (CV) test: check if two times the CV minus two times the absolute uncertainty in the CV is greater than the specification value.
            1. Calculate Coefficient of Variation (CV) for each sample in the traveling average
$$CV_{tt} = \frac{s_{tt}}{\bar{x}_{tt}}$$
            2. Calculate uncertainty in the CV for each sample in the traveling average:
                a. Relative uncertainty in s: $\frac{unc_{s,tt}}{|s_{tt}|} = \frac{1}{s_{tt}\sqrt{2*(n_{tt}-1)}}$
                b. Relative uncertainty in the average: $\frac{unc_{\bar{x},tt}}{|\bar{x}_{tt}|} = \frac{s_{tt}}{\bar{\bar{x}}_{tt}\sqrt{n_{tt}}}$
                c. Uncertainty in CV

$$\frac{unc_{CV,tt}}{|CV_{tt}|} = \sqrt{\left(\frac{unc_{s,tt}}{|s_{tt}|}\right)^2 + \left(\frac{unc_{\bar{x},tt}}{|\bar{x}_{tt}|}\right)^2}$$

v. Chi^2 test: Calculate Chi^2 to test the hypothesis that the variance is less than the specification value.
   1. H_0: s^2 <= specification, H_A: s^2 > specification

$$\nu_{tt} = n_{tt} - 1,\ s_{0,tt}^2 = \left(\frac{spec}{2.0}\bar{x}_{tt}\right)^2,\ \alpha = 0.05$$

$$\chi_{tt}^2 = \frac{\nu_{tt} s_{tt}^2}{s_{0,tt}^2}$$

   $\chi_{\alpha,\nu tt}^2$= chisqr_cvf(α, ν_{tt}) [IDL command for Chi^2 95% value.]
   2. H_0 passes if $\chi_{tt}^2 < \chi_{\alpha,\nu tt}^2$ and the band-average is not out-of-spec.

vi. Write traveling average and test results to database table for each time sample

$$\bar{x}_{tt}, \sigma_{tt}, n_{tt}, CV_{tt}, unc_{CVtt}, \chi_{tt}^2, \chi_{\alpha,\nu tt}^2$$

3. For each tt, set the valid flag to Y for the latest run. Set the valid flag to N for previous runs.
4. Generate plots of CV and Chi^2 statistics over time. Templates are below.

4.5.5 OLI Non-linear Response Characterization

4.5.5.1 Background/Introduction

The output of the OLI instrument is quantized output (Q) in units of digital number (DN). This Q is expected to be related to the input signal of the detectors, but that relationship may not be linear. Each detector may have unique non-linear irregularities in response that must be corrected in processing.

Figure 4-99 shows a response slope for a typical detector from Band 1, SCA 1. The integration time is linearly related to input radiance and is normalized to the radiance setting of the integrating sphere (DSS) used in the Ball Aerospace radiometric test collections. It can be seen that the detector response is very linear within its dynamic range. It is expected that non-linear behavior occurs just before the high and low saturation levels. There is some non-linear behavior, however, even in the center of the response -- Figure 4-100. Linear Fit Residuals shows a plot of the residuals of a linear fit made within the detector's dynamic range. All detectors studied exhibit this type of behavior.

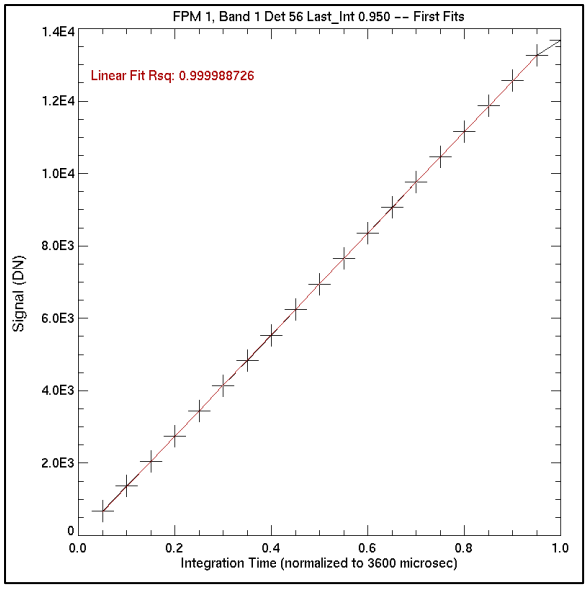

Figure 4-99. Detector Response Slope

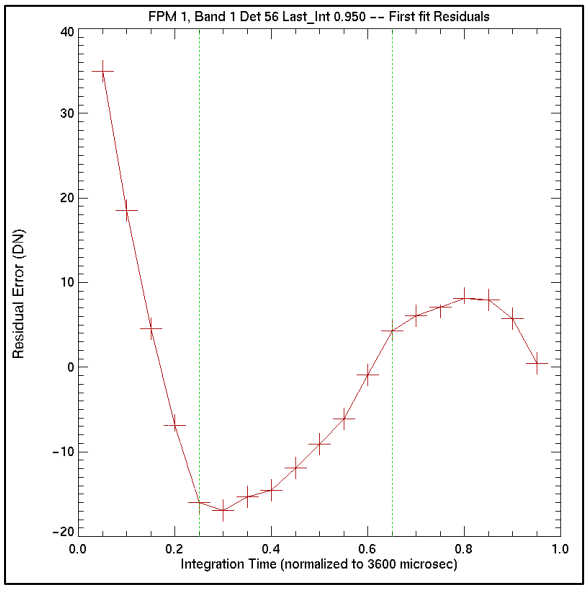

Figure 4-100. Linear Fit Residuals

From test collections made either prelaunch or in orbit, a set of parameters can be derived to linearize the detector response. The preferred data should be Integration Time Sweep (ITS) collections made with the diffuser panel as a background.

The intended form of the linearization equation is piecewise quadratic, with three distinct regions. The cutoff points between the regions – the points where the functions intersect – are determined by equating the adjacent functions. The first (bottom) region extends from zero up to the first minimum of the linear fit residual plot. The second (middle) region extends up to the last experimental point within the detector's dynamic

range. The third (top) region covers the most non-linear portion of the detector response – from the top of the detector's dynamic range to the high analog saturation point.

4.5.5.2 Inputs

Description	Symbol	Units	Level	Source	Type
L0 (Bias corrected) Mean for each integration time collection	Q	DN	N_{band} x N_{det} x N_{level}	Db (Histogram)	Float
Radiance level for each integration time collection	R	W/m^2 sr µm	N_{level}	Db (Histogram)	Float
Integration time for each collection	i		N_{level}		Float

4.5.5.3 Outputs

Description	Symbol	Units	Level	Target	Type
Remapping function cutoff thresholds		DN	N_{band} x N_{SCA} x N_{det} x N_{cutoff}	CPF	Float
Remapping function parameters			N_{band} x N_{SCA} x N_{det} x N_{coeff} x ($N_{cutoffs+1}$)	CPF	Float
Mean absolute residual		DN	N_{band} x N_{SCA} x N_{det}	Report file	Float
Maximum residual		DN	N_{band} x N_{SCA} x N_{det}	Report file	Float

Note: N_{cutoff} equals 2 in the current implementation.

4.5.5.4 Options

The cutoff point table below (in Procedure Section 2.1) contains an array of floating point values that are input to the work order but that should be editable by the operator.

4.5.5.5 Procedure

1.0 Prepare Diffuser Data

Each integration time sweep collection for OLI will involve a Diffuser collection taken at an integration time setting, which is given in the metadata. For each collection the mean DN value for each detector should be stored in the database by the Histogram Statistics procedure.

To prepare data for linearity characterization, the diffuser collections to use should be identified and the per-detector means obtained. For each collection the solar angle and viewed radiance for each band should be calculated. Once collated, the prepared data should be an array of [Q,R,i] ([mean DN, viewed radiance, integration time]) for every band and detector.

2.0 Linearity Characterization

2.1 Find preliminary cutoff points.

The cutoffs are initially chosen as the available effective radiances that are nearest to the radiances from Table 4-62 below:

Band	Band name	First cutoff R (W/m^2 sr µm)	Second cutoff R (W/m^2 sr µm)
1	CA	147.25	265.05
2	B	121.00	211.75
3	G	113.61	198.81
4	R	96.01	192.02
5	NIR	67.87	135.74
6	1SW	14.90	26.08
7	2SW	7.36	11.04
8	P	195.16	362.44
9	CRS	23.01	64.43

Table 4-62. OLI Linearity Cutoff Radiances

The effective radiance of each integration time sweep collection is calculated:

$R(i) = R_{max}*i$

The collection whose effective radiance is nearest to the first cutoff point is labeled i_{low}:

$Q_{low} = Q(d, i_{low})$
$R_{low} = R(i_{low})$

The collection with effective radiance nearest to the second cutoff point is labeled Q_{high} (with corresponding R_{high} and i_{high}).

These labels are made once for each band. These 'a priori' radiance cutoff levels should be work order parameters that are adjustable by the IAS analysts.

2.2 Create ideal line

Once the cutoff points have been found, the ideal line for each detector can be calculated. On OLI, linear behavior is assumed to pass through the origin (0 DN, 0 Radiance). The slope of the ideal line is then:

$M(d) = Q_{high}/ R_{high} = Q(d, i_{high})/(R_{max}*i_{high})$

An ideal line curve is then created, with one point for every integration time:

$Q_{ideal}(d,i) = M(d)*R_{max}*i$

2.3 Polynomial fit

The parameters of the linearization are then found by fitting the actual data to the ideal line. This is done in three regions: 0 to i_{low}, i_{low} to i_{high}, and i_{high} to the highest available effective radiance.

The fit is currently done with the IDL routine POLY_FIT, which uses the matrix inversion method that can be found in Numerical Recipes. Any implementation of a two-dimensional polynomial fit can be used. The output of each fit for each region should be quadratic coefficients such that:

$$Q'(d,r,i) = c(r,2)*Q(d,r,i)^2 + c(r,1)*Q(d,r,i) + c(r,0) \approx Q_{ideal}(d,r,i)$$

where Q'(d,r,i) = The linearized DN values (one for every i).
c(r,n) = The *nth* order quadratic coefficient for region *r*.

2.4 Determine actual cutoff points.

The actual cutoff points should be calculated after the quadratic coefficients, so that the fit between regions is as smooth as possible. The cutoff points are calculated by finding the point at which the fitted curves for two regions meet. This point can be calculated with the quadratic equation:

$$Q'_{low} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

where a = The difference in the second-order coefficients for regions 1 and 2.
a = c(2,2) – c(1,2)
b = The difference in the first-order coefficients for regions 1 and 2.
b = c(2,1) – c(1,1)
c = The difference in the zeroth-order coefficients for regions 1 and 2.
a = c(2,0) – c(1,0)

The quadratic equation has two solutions, due to the ± in the equation, and neither solution is guaranteed to be a real or positive number. Both solutions should be calculated, and the real component compared to the 'a priori' cutoff point. The solution that is closest to the first 'a priori' cutoff value Q_{low} should be chosen as the final cutoff value.

Similarly, Q'_{high} is calculated between the second and third regions (c(3,n) – c(2,n)), and its two solutions are compared to the second 'a priori' cutoff, Q_{high}.

2.5 Report output.

After the parameters have been created, the data should be linearized using them and residuals to the ideal line should be calculated. This array of residuals is used by the IAS analyst to check the accuracy of the linear characterization.

$Residuals(d,i) = Q'(d,i) - Q_{ideal}(d,i)$

The mean and maximum absolute residual for each detector should be reported in an output file.

4.5.6 OLI Response Linearization

4.5.6.1 Background/Introduction

The OLI detectors will have a non-linear relationship between the radiance viewed and the DN value output by the detectors. Because of this, the response of the detectors must be linearized as part of radiometric calibration.

The correction for non-linearity involves remapping bias corrected detector response data onto a linear curve. This is done in two stages: one linearity correction, and one per-detector Non-Uniformity Correction (NUC). In practice these corrections are similar and use the same algorithm (but different lookup tables (LUTs)), making linearity correction a two-step process.

The standard response linearization procedure uses a quadratic equation, but the instrument vendors have delivered correction parameters as full-sized lookup tables that cover all possible values in 14-bit space. The alternate response linearization algorithm is intended as an implementation of the vendor algorithms, with the sole difference that the very large vendor-provided LUTs have been compressed for performance reasons into piecewise continuous arrays that contain paired input and output values. The piecewise continuous LUT provides the same results as the full-sized LUT to within 0.001 DN, but take up approximately one-thirtieth of the disk space.

4.5.6.2 Inputs

Description	Symbol	Units	Level	Source	Type
Scene (L0rc2 - bias corrected)	*Q*	DN	N_{band} x N_{SCA} x N_{det} x N_{frame}		Float
Artifact Mask	*AM*	Unitless	N_{band} x N_{SCA}		Int
Linearity Correction LUT	*LIN*	Unitless	N_{band} x N_{SCA} x N_{det} x N_{value}	RLUT file (an extension of the CPF)	Float
Uniformity Correction LUT	*NUC*	Unitless	N_{band} x N_{SCA} x N_{det} x N_{value}	RLUT file (an extension of the CPF)	Float

4.5.6.3 Outputs

Description	Symbol	Units	Level	Target	Type
Scene (L0rc2 - bias corrected, linearized)	*Q**	DN*	N_{band} x N_{SCA} x N_{det} x N_{frame}		Float

4.5.6.4 Options

- Apply Linearity correction (default on)
- Apply Non-uniformity correction (default on)

4.5.6.5 Procedure

Response linearization should be a simple replacement of the incoming value with a linearized response value.

For each band there are two LUTs that are condensed versions of the LUTs provided by Ball. Each LUT has a pair of values – an index value and an output adjustment value – for every SCA and detector. The first LUT is intended to correct for SCA-wide non-linearities. Ball refers to this as the 'linearity' correction (abbreviated as LIN). The second LUT is intended to correct per-detector non-linearities. Ball refers to this as the 'nonuniformity' correction, or NUC. Despite their distinct names, both are currently per-detector corrections using the exact same algorithm.

Whenever alternate response linearization is performed, both the LIN and NUC corrections are run. The options to turn off one or the other correction exist only as a means to debug the RLUTs.

A search is made in INPUT LUT to find the index in the list whose values bracket the incoming DN value. Those bracketing indices are then used to get the bracketing values from the VAL LUT. The adjustment is a scaled interpolation between those two bracketing adjustment values. The output value (DN*) is the input DN plus the calculated adjustment.

The paired LUTs are as follows:

INPUT[band, sca, det, index]
VAL[band, sca, det, index]

Although these LUTs are dependent upon band, sca, and det, they are referred to as INPUT[index] and VAL[index] below for clarity. Note also that the size of the LUTs varies between detectors. The INPUT and VAL array are always the same size as each other, but the arrays used by one detector may not be the same size as the arrays used by any other detector. The maximum size of these arrays is fixed at 1000 by the compression algorithm (described below).

With

index = The largest index into the INPUT array where INPUT[index] ≤ DN.

(The values INPUT[index] and INPUT[index+1] should bracket the incoming DN value. Finding this index will entail a quick search through the INPUT array.)

Then the adjustment is an interpolation between the LUT values:

*Adj = VAL[index] + (VAL[index+1] - VAL[index]) * (DN - INPUT[index]) / (INPUT[index+1] - INPUT[index])*

DN = DN + Adj*

This process should be used for the LIN LUT and then the NUC LUT, for every pixel in an image.

Note that because index+1 is used in the algorithm, pixels whose values equal the 14-bit integer maximum (16383) should not be processed. This is far beyond saturation in most bands and should not be an issue.

The specific algorithm that compresses the vendor-provided LUTs into the INPUT and VAL LUTs is not important to the alternate response linearization algorithm, but it does dictate the size of the LUTs. Any compression algorithm may be used, as long as it results in LUTs of the same format and with the same fidelity to the original LUTs.

To clarify , the procedure for the compression algorithm is as follows:

For each detector in each band, the first non-zero adjustment in the vendor LUT (VLUT) is found as index i_0. Between the indices i_0 and i_0+1, the adjustment changes by a delta adjustment (ΔA). The linear equation describing the adjustment is then:

$$Adj(i) = (i - i_0) * \Delta A + VLUT(i_0)$$

The index *i0* is placed in the INPUT LUT, and the VLUT adjustment at that index is placed into the VAL LUT.

This equation is checked against the VLUT value at subsequent indices (*i+1*, *i+2*...) until the difference between this extrapolated adjustment and the actual VLUT adjustment is greater than the threshold value (currently set at 0.001 DN). When the threshold is exceeded, the linear extrapolation is then reset; a new i_0 is set at the index where the threshold was exceeded and becomes the next entry in the INPUT LUT, while the VLUT adjustment at the new i0 becomes the next entry in the VAL LUT. Then a new delta is calculated, and the iterations resume with the new extrapolation.

At the end of the process, a final point is placed into the arrays; the index 16,383 and the associated adjustment value at that index. This ensures that the linearization algorithm will always find two indices to bracket any incoming DN value.

This compression algorithm is run for both the LIN and NUC LUTs provided by the vendor, for every fpm, band, and detector. The output becomes the RLUTs used by the alternate response linearization algorithm.

4.5.7 OLI Relative Gain Characterization (Solar Diffuser)

4.5.7.1 Background/Introduction

This algorithm calculates the per-SCA and per-detector relative gains for each solar diffuser acquisition. The solar diffusers are the most uniform targets available to the OLI. Even still, there is some systematic non-uniformity due to illumination and view angles that result in a gradient across the panel. This algorithm assumes that the non-uniformity can be characterized and is then stable over the lifetime of the OLI.

The Solar Diffuser Characterization algorithm also calculates detector relative gains based on the diffuser data, but as those are calculated for a different purpose, they are not suitable for determining operational relative gains. The relative gains in the SOLAR_RESPONSE_DETECTOR table do not have the non-uniformity scale factors applied. The relative gains calculated by this algorithm should populate a new table.

This algorithm should be run by default in L1R-Solar OLI process flow, though this algorithm does not need to operate on image data. The existing tools use the data in the Histogram Statistics table.

Analysis of SWIR-band jumpers, acquisition-to-acquisition changes in relative gain, tracking of best and worst behaved detectors are all assessments that come out of this dataset. But these analyses will have to be implemented outside the IAS as the two algorithms developed to generate the CPF gains will not be set up to do per-acquisition time series assessments.

4.5.7.2 Dependencies

Histogram statistics characterization

4.5.7.3 Inputs

Description	Symbol	Units	Level	Source	Type
dark-subtracted scene_pixel_mean	*Q*	DN	N_{band} x N_{SCA} x N_{det}	DB Histogram Statistics, histogram_statistics_cal	Float
diffuser (working/pristine)			1	DB solar_response.diffuser_collect_type	int
absolute gains	*G*	counts /rad	N_{band}	CPF	Float
solar diffuser nonuniformity scale factors	*nu*	[]	N_{band} x N_{SCA} x N_{det}	CPF	Float

4.5.7.4 Outputs

Description	Symbol	Units	Level	Source	Type
diffuser (working/pristine)			1	DB SOLAR_*_RELGAINS	int

Description	Symbol	Units	Level	Source	Type
per-acquisition relative gains (detector)	r_d	[]	N_{band} x N_{SCA} x N_{det}	DB SOLAR_DETECTOR_RELGAINS	Float
per-acquisition relative gains (module)	r_m	[]	N_{band} x N_{SCA}	DB SOLAR_MODULE_RELGAINS	float
valid flag		[]	1	DB SOLAR_*_RELGAINS	int

4.5.7.5 Procedure

For each nominal solar acquisition (working and pristine), for each band:

1. Extract data from the Histogram Stat (Cal) table for the acquisition; Q_i. Extract gains and uniformity scale factors from the CPF: G and nu. Note that nu is diffuser specific.
   a. The correct processing level for the data is: histogram_statistics.processing_level = 'CTBL'
2. Correct signal for diffuser non-uniformities and known FPM-to-FPM discontinuities (via per-FPM gains). This isn't quite radiance, since we are not correcting for relative gains, but it has the units of radiance.
   a. $\widetilde{L_i} = \frac{Q_i}{G_i * nu_i}$
3. Calculate the average radiance for each module.
   a. $\widetilde{L_{m(J)}} = average(\tilde{L}_{j,i})$
   b. where *i* includes the detectors from the *j*th module.
4. Calculate module-average detector relative gains
   a. $r_{d(i)} = \frac{\widetilde{L_i}}{\widetilde{L_{m(J)}}}$
   b. where *i* is the detectors from the *j*th module
5. Calculate module relative gains
   a. $for\ j = 0,\ n_{fpms}\ do\ r_{m(j)} = \frac{\widetilde{L_{m(J)}}}{mean(\widetilde{L_m})}$
   b. where j is the module number and i is the detector number
6. Record relgains to database tables for use by AutoCal Diffuser Relative Gains.
   a. The valid flag can be used to indicate whether or not the data are suitable for including in the averages generated by AutoCal Diffuser Relative Gains. Default for the flag is valid, but an analyst can change it to invalid if the acquisition is determined to be problematic.

4.5.8 OLI Detector Response Characterization (Solar Diffuser)

4.5.8.1 Background/Introduction

The Primary radiometric calibration devices on OLI are solar diffusers. These Spectralon panels are rotated in position in front of the OLI aperture and the spacecraft is oriented so as to bring sunlight in the solar diffuser port so that sunlight is reflected into the OLI aperture. There are 2 solar diffusers on-board OLI, a Primary and pristine. The current plan calls for the Primary diffuser to be deployed approximately weekly during normal operations. The pristine diffuser will be deployed on a less frequent basis and used to monitor degradation of the Primary panel. The OLI diffuser data provides a valuable source for deriving radiometric gain updates for L1r processing, performing instrument characterizations and monitoring uniformity and stability requirements.

Both radiance and reflectance based gains will be derived and stored. To derive radiance gains, both bias and non-linearity corrected OLI data (0Rc) values (in DN) and diffuser spectral radiance values (W/m2-sr □m) are required. With the exception of a correction for Earth-Sun distance, the diffuser spectral radiance values are relatively static as the sun reflects off the diffuser at a fixed angle and the diffusers are expected to change slowly in reflectance characteristics. The radiance values adjusted to a reference Earth-sun distance can be stored in the CPF. (Note. there may be multiple versions of these values i.e., prelaunch, postlaunch and current that could be stored in a single CPF or as separate CPFs). The Earth-Sun distance correction will be derived from JPL ephemeris data. Gains will be generated and output as per detector and SCA relative gains, and band-averaged gains. Similar outputs for reflectance-based gains will be derived based on spectral reflectance values (unitless) in place of the spectral radiance values.

Approximately 500 frames of data with the solar diffuser illuminated will be acquired per typical collect. However, there are some less frequent acquisitions that will change the size of these datasets. These collects will be acquired at variable integration times for non-linearity characterization and only the nominal integration time portion of these collects will be processed by this algorithm. A second acquisition scenario involves 60 second long solar diffuser collects; these collects are used to evaluate the short-term stability of the OLI response.

Ephemeris and attitude data will be processed and used to confirm the solar pointing during each solar acquisition. This processing may occur in part separately prior to use in this algorithm. It is assumed that all ancillary data for verification of the acquisition will come from the wideband ancillary data. Separately, the MOC will provide the CVT a report on the solar calibration maneuver indicating whether the maneuver was executed as planned, e.g., the solar pointing was as required.

To track pristine and Primary diffuser datasets a diffuser parameter needs to be identified, associated with the appropriate inputs and output parameters and stored in the trending DB. Artifacts will be accounted for in the L0rc inputs to processing.

Assessment of algorithm outputs will be performed by comparing the “original” solar radiance and reflectance calibrated products, to “new” solar products derived with the newly derived gains. While not performed by this algorithm, these product generation steps should be considered a routine part of “solar characterization process.” The evaluation of these products and reports will be performed ‘off-line’

4.5.8.2 Inputs

Description	Symbol	Units	Level	Source	Type
L0rc (Bias corrected and linearized) Means	Qnet	DN	N_{band} x N_{det}	Db (Histogram)	Float
L0rc Stdev	Qσnet	DN	N_{band} x N_{det}	Db (Histogram)	Float
Start time		GMT		Db	Float
Stop time		GMT		Db	Float
Number of lines used				Db (Histogram)	Int
Inoperable Detectors			N_{band} x N_{det}	CPF	Int
Attitude Control System Quaternion Coefficients	Q0,Q1,Q2, Q3			Ancillary Preprocessing	Fltarr
Diffuser Radiance@nominal SZA, E-S Distance (1A.U.)	Lλ	W/m^2 sr μm	N_{DT} x N_{band} x N_{det}	CPF	Float
Diffuser Deployment Angle		degrees		Ancillary File	Float
Earth –Sun Distance*	des	AU		Auxiliary File	Float
Diffuser Bidirectional Reflectance Factor	ρ		N_{DT} x N_{band} x N_{det}	CPF or Ancillary	Float
Integration Time				Header or DB	
Diffuser Type (DT) i.e., Pristine or Primary				L0R	
Start_time				L0R	
Stop_time				L0R	
Collection Type				L0r, Metadata	
Sun Zenith Angle Mean	Θs	degrees		Report, Db	
Sun Zenith Angle Stdev	Θsσ	degrees		Report, Db	Float
Sun Azimuthal Angle Mean	Φs	degrees		Report, Db	Float
Sun Azimuthal Angle Stdev	Φsσ	degrees		Report, Db	Float

*From JPL data

4.5.8.3 Outputs

Description	Symbol	Units	Level	Target	Type
Start time				Report, Db	Float
Stop time				Report, Db	Float
Lines averaged				Report, Db	Int
Detector Relative Gains Mean	G_{rel}		N_{band} x N_{det}	Db	Float
Detector Relative Gain Stdev	$Gσ_{rel}$		N_{band} x N_{det}	Db	Float

Description	Symbol	Units	Level	Target	Type
SCA Relative Gains Mean	G_{SCA}		N_{band} x N_{SCA}	Db	Float
SCA Relative Gain Stdev	$G\sigma_{SCA}$		N_{band} x N_{SCA}	Db	Float
Radiance-Band Avg Gains Mean	G	DN/ W-m^2-sr-□m	N_{band}	Report, Db	Float
Radiance-Band Avg Gains Stdev	$G\sigma$	DN	N_{band}	Report, Db	Float
Reflectance-Band Avg Gains Mean	$G\rho$	DN	N_{band}	Report, Db	Float
Reflectance-Band Avg Gains Stdev	$G\rho\sigma$	DN	N_{band}	Report, Db	Float
Diffuser Type	DT			Report, Db	Int
Deployment Angle		degrees		Report, Db	Float
View Sun Zenith Angle Mean*	Θ_v	degrees		Report, Db	Float
View Sun Zenith Angle Stdev*	$\Theta_v\sigma$	degrees	N_{band}	Report, Db	Float
View Sun Relative Azimuthal Angle Mean*	Φ_v	degrees	N_{band}	Report, Db	Float
View Sun Relative Azimuthal Angle Stdev*	$\Phi_v\sigma$	degrees		Report, Db	Float
Incident Sun Zenith Angle*	Θ_s	degrees		Report,Db	Float
Incident Sun Zenith Angle Stdev*	Θ_s	degrees		Report,Db	Float

*From JPL data

4.5.8.4 Options

Summary report to be generated with every solar processing run.

4.5.8.5 Procedure

Note, while all procedure steps will be identical for both Diffuser Types (DT) Primary panel (both nominal and Time Sweep acquisitions) and the Pristine panel, certain inputs (as identified by the input table) need to be identified/processed by its Diffuser Type (DT).

1.0 Verify solar acquisition:

1.1 Read in ancillary data and verify/output solar collect information e.g.
Diffuser Deployment Resolver Position
Collect Sequence
Integration Time
Verify the integration time is nominal.

IF not verified THEN stop processing ELSE

1.2 Derive Solar Angles:

Sun angles are calculated in the Landsat 8/9 Solar Calibration Frame of Reference system (LSCF). In the solar calibration mode, the spacecraft is aligned to the LSCF i.e., the pitch axis is aligned to the sun (-Y), and the yaw axis to the projected nadir vector (+Z). *Angle(s) should be constant over the entire acquisition with pointing control.* Figure 4-101 illustrates the vectors and angles wrt to a diffuser panel (small oval), mounted within the rotating calibration assembly (large oval).

θs = incident angle between diffuser normal and sun vector direction
θv = view (scatter) angle between diffuser normal and sensor line-of sight vector.

Both theta angles should be about 45 degrees wrt prelaunch alignment of the diffuser prelaunch that should remain constant throughout the mission.

φv = view (scatter) relative azimuthal angle between diffuser normal and sensor line-of sight

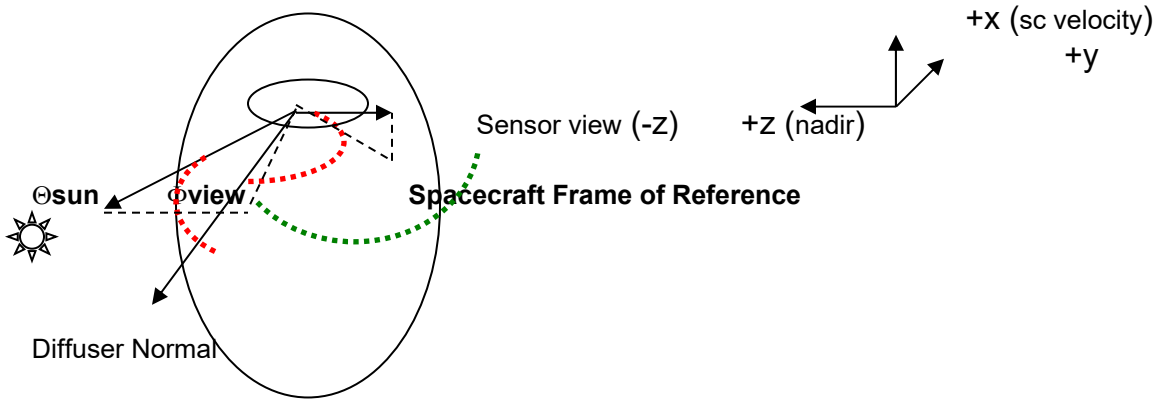

Figure 4-101. Diffuser Solar Angles Defined wrt Diffuser Normal Component

1.2.1 Process S/C Attitude: Extract spacecraft (ACS) quaternion coefficients from ancillary data preprocessing, corresponding to calibration interval.

1.2.2 Retrieve JPL Solar Ephemeris i.e., sx,sy,sz

For all Attitude quaternion values in solar cal acquisition

1.2.3 Transform sun vectors from Earth-centered to LSCF for all ephemeris points in acquisition.

Xs,Ys,Zs = TR * sx,sy,sz using values from 1.2.1 and 1.2.2

Where:

TRansform Matrix =
TR(0,0) = 1-2*(q2(i)^2+q3(i)^2)
TR(1,0) = 2*(q1(i)*q2(i)+q0(i)*q3(i))
TR(2,0) = 2*(q1(i)*q3(i)-q0(i)*q2(i))
TR(0,1) = 2*(q1(i)*q2(i)-q0(i)*q3(i))
TR(1,1) = 1-2*(q1(i)^2+q3(i)^2)
TR(2,1) = 2*(q0(i)*q1(i)+q2(i)*q3(i))
TR(0,2) = 2*(q0(i)*q2(i)+q1(i)*q3(i))
TR(1,2) = 2*(q2(i)*q3(i)-q0(i)*q1(i))
TR(2,2) = 1-2*(q1(i)^2+q2(i)^2)

1.2.4 Derive array of sun angles using transformed sun components i.e., Xs, Ys, Zs,

panel orientation components i.e.,
Xn = 0.
Yn = -cos(45.)
Zn = -sin(45.)

and view direction components i.e.
Xv = 0.
Yv = 0.
Zv = 1

Incident Solar Zenith Angle

cosi = Xn*Xs + Yn*Ys+Zn*Zs
sini = sqrt (1-cosi^2)
ϑs= atan(sini/cosi)

View Solar Zenith Angle

cosi = atan (Xn*Xv + Yn*Yv+Zn*Zv)
sini = sqrt (1-cosi^2)
ϑv = atan(sini/cosi)

View Solar Relative Azimuthal Angle

cosa = Xns*Xnv+Yns*Ynv+Zns*Znv
sina = sqrt(1-cosa^2)
φv = atan (sina/cosa)

Where Xns,Yns,Zns = sun normal , the cross product of XnYnZn with XsYs,Zs i.e.

Xnv,Ynv,Znv = view normal , the cross product of

XnYnZn with XYZ-view direction components i.e., the only non-zero view component (zv=1) along the sensor view axis.

End Attitude loop

1.2.5 Derive sun angle means and standard deviations

$\Theta s = \text{mean}(\vartheta s)$ & $\Theta s\sigma = \text{Stdev}(\vartheta s)$
$\Theta v = \text{mean}(\vartheta v)$ & $\Theta v\sigma = \text{Stdev}(\vartheta v)$
$\Phi v = \text{mean}(\varphi v)$ & $\Phi v\sigma = \text{Stdev}(\varphi v)$

2.0 DERIVE RADIANCE GAINS:
FOR each Band:

2.1 Read per detector spectral radiances - $L_\lambda (b,d)$

2.2 For date of acquisition, read in appropriate value of Earth-Sun distance, d_{es}

FOR each detector

2.3 Read in L0rc means ($<Q_{net}(b,d)>$) and standard deviations ($Q\sigma_{net}(b,d)$) derived from histogram statistics

2.4 Derive absolute per detector gain mean and standard deviation, i.e.

$$G(b,d) = d_{es}^{\,2} * <Q_{net}(b,d)> / L_\lambda(b,d) \quad (3)$$
$$G\sigma(b,d) = d_{es}^{\,2} * Q\sigma_{net}(b,d) / L_\lambda(b,d) \quad (3a)$$

END detector loop

2.5. Generate/output the band mean gain and standard deviation (over all operable detectors)

$$G_{aBands}(b) = < G(b,*)> \ \& \ G\sigma_{aBands}(b) = \text{Stdev}(G(b,*)) \quad (4)$$

2.6 Generate/output the per detector mean relative gain and standard deviation , wrt to the per band average.

$$G_{rel}(b,d) = G(b,d) / G_{aBands}(b) \ \&$$
$$G\sigma_{rel}(b,d) = G\sigma(b,d) / G_{aBands}(b) \quad (5)$$

END band loop

2.7 Generate/output the per SCA mean relative gain and standard deviation , wrt to the per band average.

$G_{SCA}(b,SCA) = \langle G(b,SCA) \rangle / G_{aBands}(b)$ &
$G\sigma_{SCA}(b,SCA) = \langle G\sigma(b,SCA) \rangle / G_{aBands}(b)$ (6)

3.0 DERIVE REFLECTANCE GAINS:
FOR each Band
FOR each Detector:

3.1 Read per detector bidirectional reflectance factors $\rho(b,d)$

3.2 Derive absolute per detector average reflectance gains $G\rho(b,d)$, and per detector reflectance gain standard deviations $G\rho\sigma(b,d)$ = i.e.

$$G\rho(b,d) = (Q_{net}(b,d) * d_{es}^{2}) / (\rho(b,d) * \cos(\Theta_s)) \quad (7)$$
$$G\rho\sigma(b,d) = (Q\sigma_{net}(b,d) * d_{es}^{2}) / (\rho(b,d) * \cos(\Theta s)) \quad (7a)$$

Where $Q_{net}(b,d)$ = per detector diffuser corrected response
$\rho(b,d)$ = diffuser reflectance factors
Θ_s = solar zenith diffuser angle (nominally 45°)
d_{es} = Earth-Sun distance

End detector loop

3.3 Generate/output the band mean gain and standard deviation (over all operable detectors)

$$G\rho_{aBands}(b) = \langle G\rho(b,*) \rangle \text{ and } G\sigma_{aBands}(b) = \text{Stdev}(G\rho(b,*)) \quad (8)$$

End bands

4.0 Generate summary report for every processing run with hardcoded formats. (Example format)

SOLAR CALIBRATION SUMMARY REPORT

Report Date
Start Acq Time
Stop Acq Time
Total Acq Time

Diffuser Type: (i.e., Pristine, Primary)
Collect Sequence Type: (i.e., Nominal, Integration Time Sweep)
Integration Time:

Solar Distance and Diffuser Angles:
E-S Distance (AU)

Deployment Angle (degs)
Solar Zenith Angle (Theta-i, degs)
View Zenith Angle (Theta-v, degs)
Sun Azimuth Angle (Phi-v, degs)
Ephem Outliers:

Image Stats (Bias Corrected & Linearized)
BandID Avg Stdev

Diffuser Radiance & Diffuser Gains
BandID Avg Avg Stdev

Diffuser Reflectance & Reflectance Gains
BandID Avg Avg Stdev

Total Frames Processed

4.5.9 OLI Standalone 60 Second Radiometric Stability Characterization

4.5.9.1 Background/Introduction

Both the bias and the gain stability of an instrument are contributing factors to variability within a radiometrically calibrated product. The OLI has a “60 second stability,” a.k.a. short-term stability requirement, specifically designed to control this within-product variability. The specific requirement states: “Over any time period up to 60 seconds, after radiometric correction, the scene-averaged OLI image data for radiometrically constant targets with radiances greater than or equal to $L_{typical}$ shall not vary by more than plus or minus 0.5% (95% or 2 sigma confidence interval) of measured radiance.” The 60-sec stability characterization requires scenes of temporally uniform radiance above $L_{typical}$, e.g., long solar acquisitions. Long shutter data will also be analyzed to allow determination of the bias stability contribution to the overall stability. In the processing flow, these collects should be regarded as “special” collects vs. “standard” dark or solar collects identifying the data collects types that will meet the scene(s) minimum length criteria required for this processing.

As reflected in the OLI Ops-Con, special 60 second long solar acquisitions and related 60 second long dark collects, made at the nominal integration time, are the initial data source for this algorithm. They are processed through the Histogram Statistics Characterization to generate the input to this algorithm. This short-term stability characterization will produce four categories of outputs:

1. Dark Shutter Raw data SCA level stability in DN
2. Solar Diffuser bias and non-linearity corrected SCA level stability in DN
3. Solar Diffuser SCA level stability in %
4. Solar Diffuser Detector level stability in %

The individual detector relative gain short-term stability will be used in the analysis for the non-uniformity requirement stability and track actual performance characteristics against the system engineering allocations.

Solar radiance histograms with absolute and relative gains applied will be used to directly verify the 0.5% radiometric stability requirement. The solar radiance stability metric will be converted from radiance units to % so it can be evaluated directly against the 0.5% requirement. Solar data that is bias and non-linearity corrected will be used to characterize the 60s band mean and relative gains stabilities.

Note that the bias used for bias removal is derived from a model that is based on the pre and post solar collect event dark shutter collects which are also taken at the nominal integration time.

4.5.9.2 Inputs

Description	Symbol	Units	Level	Source
Signal Mean (Solar Diffuser Only)	$\overline{Q_{B,S,D}}$	DN	N_{band} x N_{SCA} x N_{det}	Histogram Stat Char
Signal StDev	*Sigma_*$Q_{B,S,D}$	DN	N_{band} x N_{SCA} x N_{det}	Histogram[Error! Bookmark not defined.] Stat Char
Scene Type (i.e S or D)	Stype	Char	N_{band}	DB L0Rp Image file
Diffuser Type (i.e. W or P) (Solar Diffuser Only)	Dtype	Char	N_{band}	DB L0Rp Image file
Inoperable Detector List	Dinop	Float	N_{band} x N_{SCA}	CPF

4.5.9.3 Outputs

Description	Symbol	Units	Level	Target
SCA level Stability in DN	$\overline{\Delta Q_{B,S}}$	DN	N_{band} x N_{SCA}	DB (Bias & Solar Stability)
SCA Level Stability in %	$\overline{\Delta_{B,S}}$	%	N_{band} x N_{SCA}	DB (Gains & Solar Radiance Stability)
Detector Level Stability in %	$\overline{\Delta_{B,S,D}}$	%	N_{band} x N_{SCA} x N_{det}	DB (Solar)
Diffuser Type (i.e W or P) (Solar Diffuser Only)	Dtype	Char	N_{band}	DB

4.5.9.4 Options

Trending On/Off Switch: If trending is Off, output parameters are written to a text file.

4.5.9.5 Procedure

Note that in order to characterize radiometric stability we call this algorithm at least 2 times: Once to process long dark shutter histogram data, and once to process the histogram data for the closest long solar collect.

The trend IDs for long dark and long solar collects processing need to be found. Based on these trend ID's and the processing level the correct histograms information can be imported to this algorithm.

Calculating statistics (get it from Hits Stat)

---

1. Based on the inoperable detector list, omit those detectors when calculating SCA level averages.
2. Calculate the average across all detectors within an SCA.

$$\bar{Q}_{B,S} = \bar{Q}_{B,S,D} \quad (1)$$

Calculating Stability of Signal

---

3. For each detector, calculate the 2-sigma along-track variation in the image that is at the length of Δt=60sec.
    a. $\Delta Q_{(B,S,D)} = 2 \times Sigma\_\overline{Q}_{(B,S,D)}$ (2)

Processing data to generate outputs

---

4. Calculate the SCA average variability, i.e, the average across all detectors within each SCA:
    a. $\Delta Q_{B,S} = \overline{\Delta Q_{B,S,D}}$ (3)
5. Record the SCA mean variabilities () to the database or output file (along with other specified outputs in the output table). This is the end of the algorithm for dark data.
6. For solar data also do this:
    a. Calculate the per-detector variability in terms of percent change. It illustrates the impact of individual detectors on the overall radiometric stability.
        i. $\Delta_{B,S,D} = \overline{\left(\frac{\Delta Q_{B,S,D}}{\bar{Q}_{B,S,D}}\right)} * 100\%$ (4)

b. Calculate the SCA average variability in terms of percent gain change:

i. $$\Delta_{B,S} = \overline{\left(\frac{\Delta Q_{B,S}}{\bar{Q}_{B,S}}\right)} * 100\% \quad (5)$$

c. Write the per-detector percent variability ($\Delta_{B,S,D}$) and the SCA average variability ($\Delta_{B,S}$) to the database or output file (along with other specified outputs in the output table).

4.5.10 OLI Detector Response Characterization (Lamp)

4.5.10.1 Background/Introduction

The OLI will have 2 stimulation lamp fixtures. Each fixture will contain 6 tungsten lamps. A lamp in each fixture is paired (wired in series with) a lamp in the other fixture. Of the 6 lamp pairs, 3 are designated primary (operate on side A electronics) and 3 redundant (operate on side B electronics). Only one lamp pair is to be powered at a time. One pair of lamps on each side of the electronics with be designated “WORKING” with data to be acquired on a daily basis once operational. One pair of lamps will be designated “BACKUP” with nominally monthly usage and one pair “PRISTINE” with semi-annual usage. Use of any redundant lamp can only occur when the full instrument is switched to the B-side electronics.

The current OLI concept of operations stipulates that the on-board lamps will be exercised daily. Data acquisition at the typical line rate of 4.236 ms/frame will begin several seconds prior to lamps turn-on, proceed with 2.75 minutes of lamp data, and continue until several seconds after the lamps have been turned off, yielding a total of approximately 3.34 minutes of data. However, the intent of the present algorithm is to provide trending statistics for characterizing post lamp warm-up detector responses. Therefore this algorithm will operate on 512(default value) lines of data taken 2.5 minutes after lamps turn-on and populate the database with the following statistics:

a) the mean, standard deviations, skewness and kurtosis at the detector level,
b) the averaged mean and standard deviation per band per sensor chip array (SCA) level, and
c) the averaged mean and standard deviation per band

The full 2.75 minutes of daily lamp data may be used for further lamp stability studies, or for assessment of FPA, Focal Plane Electronics (FPE) or Lamp sub-system performances. Multiple within-orbit lamp collects separated by off time may also be prescribed for special study. Analysis of these extended datasets and lamp collects will be conducted off line as they are beyond the scope of the present algorithm. Associated telemetry from the stimulation lamps may be required to support the detector characterizations, for example:

Lamp Housing Temperatures (2), Monitoring diode outputs (2), Lamp currents, Lamp voltages, monitoring diode temperatures (2), selected lamp, lamp on-time. These telemetry data points will be included in the ancillary data within the wideband data.

The initial effective lamp radiance for each detector (to be provided by BATC) will be available in the CPF. Using the appropriate lamp telemetry parameters, these will be corrected for the on-board conditions during the lamp imaging event and the result stored in the database for comparison with the Level1R statistics.

4.5.10.2 Inputs

Description	Symbol	Units	Level	Source	Type
Scene/Interval ID		unitless	1	dB (ancillary or MOC report)	Text
Bias and non-linear corrected (L0rc)		DN	N_{band} x N_{SCA} x N_{det} x N_{frame}		Integer
Lamp Level1		Scaled DN or radiance units	N_{band} x N_{SCA} x N_{det} x N_{frame}		Integer or Float
Impulse Noise			N_{band} x N_{SCA} x N_{det}	LM	Integer
Saturated Pixels			N_{band} x N_{SCA} x N_{det}	LM	Integer
Inoperable Detectors			N_{band} x N_{SCA} x N_{det}	CPF	Integer
Dropped Frames			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Integer
Selected Lamp Pair		unitless	1	dB (ancillary)	Integer
Data Start Time		UTC	1	dB (ancillary or MOC report)	*
Lamp on/off time		UTC	1	dB (ancillary or MOC report)	*
Start and number of lines to process Statistics start time (Time offset from lamp turn on)		Frame number	2	ODL or dB(ancillary or MOC report)	Integer
Statistics data block size (nominally 512)			1	ODL or dB	Integer
Data Frame rate		Samples per second	1	dB (ancillary or MOC report)	Float
Lamp current		Amps	$N_{telemetry}$	dB (ancillary or MOC report)	Float
Lamp voltage		Volts	$N_{telemetry}$	dB (ancillary or MOC report)	Float
Monitor voltage		Volts	$N_{telemetry}$	dB (ancillary or MOC report)	Float
Lamp Housing Temperature		degrees Celsius	$N_{telemetry}$	dB (ancillary or MOC report)	Float
Monitor Diode temperatures		degrees Celsius	$N_{telemetry}$	dB (ancillary or MOC report)	Float
Focal Plane Temperatures		Kelvins	2	dB (ancillary or MOC report)	Float

* UTC in hhmmss.sss

4.5.10.3 Outputs

Description	Symbol	Units	Level	Target	Type
Scene/Interval ID		unitless	1	dB	Text
Selected Lamp Pair		unitless	1	dB	Integer
Lamp on Time		Seconds	1	dB	Float
Processing interval:					
Start/Stop Time		UTC		dB	*
Start/Stop Frame/Line Numbers		unitless		dB	Integer
Lamp Level0c statistics (mean, stddev, skewness, kurtosis): a. Per Band per SCA per Detector b. Per Band per SCA c. Per Band		DN	a) N_{band} x N_{SCA} x N_{det} b) N_{band} x N_{SCA} c) N_{band}	dB	Float
Lamp Level1R statistics (mean, stddev, skewness, kurtosis): a. Per Band per SCA per Detector b. Per Band per SCA c. Per Band		DN	a) N_{band} x N_{SCA} x N_{det} b) N_{band} x N_{SCA} c) N_{band}	dB	Float
Lamp Current A:					
Mean		Amps	1	dB	Float
Sigma		Amps	1	dB	Float
Lamp Voltage A:					
Mean		Volts	1	dB	Float
Sigma		Volts	1	dB	Float
Monitor Voltage A:					
Mean		Volts	1	dB	Float
Sigma		Volts	1	dB	Float
Lamp Housing Temperature A:					
Mean		Degrees Celsius	1	dB	Float
Sigma		Degrees Celsius	1	dB	Float
Monitor Diode Temperature A:					
Mean		Degrees Celsius	1	dB	Float
Sigma		Degrees Celsius	1	dB	Float
Lamp Current B:					
Mean		Amps	1	dB	Float

Description	Symbol	Units	Level	Target	Type
Sigma		Amps	1	dB	Float
Lamp Voltage B:					
Mean		Volts	1	dB	Float
Sigma		Volts	1	dB	Float
Monitor Voltage B:					
Mean		Volts	1	dB	Float
Sigma		Volts	1	dB	Float
Lamp Housing Temperature B:					
Mean		Degrees Celsius	1	dB	Float
Sigma		Degrees Celsius	1	dB	Float
Monitor Diode Temperature B:					
Mean		Degrees Celsius	1	dB	Float
Sigma		Degrees Celsius	1	dB	Float
Selected Lamp			1	dB	Integer
Focal Plane Temperatures at FPM7 and FPM14:				dB	Float
FPM 7 Mean		Kelvin	2	dB	Float
FPM 7 Sigma		Kelvin	2	dB	Float
FPM 14 Mean		Kelvin	2	dB	Float
FPM 14 Sigma		Kelvin	2	dB	Float

* UTC in hhmmss.sss

4.5.10.4 Options

Lamp warm-up interval input – Specifies Time (UTC) and image line number interval allowed for lamp warm-up.

Report outputs – Enables output of statistics results and telemetry data (e.g., Temperatures, diode voltages) in text format.

4.5.10.5 Procedure

I. Read L0c and Level1R data.

   a. Calculate frame number at 2.5 minutes (stats_starttime) from lamp turn on, and frame number at end of statistics data block (default stats_nsamples=512):

      i. Stats_startframe=framerate*stats_starttime

      ii. Stats_endframe=stats_startframe+stats_nsamples

b. Extract data block from stats_startframe to stats_endframe from the appropriate data files.

II. For L0c and Level1R lamp statistics data block do:

a. Calculate statistics for each detector (det), band, SCA.

i. Initialize statistics arrays:

$$Mean\ (\bar{x}), S\tan dard\ deviation\ (\sigma), Skewness\ (S), Kurtosis\ (k) = float\ (n\det\_\ ms\ or\ n\det\_\ pn, nBands, nSCA), where\ \ nBands = 9,$$
$$nSCA = 14, n\det, n\det_{ms} = 494, n\det_{pn} = 988$$

ms denote ms bands, *pn* denotes pan band

ii. For each detector, Band, SCA compute:

1. Mean, $\bar{x}_{(\det,band,SCA)} = \frac{1}{nFrames}\sum_{i=StartFrame}^{EndFrame} x_i$,
$where\ \ nFrames = EndFrame - StartFrame$

2. Standard Deviation, $\sigma_{(\det,band,SCA)} = \sqrt{\frac{1}{nFrames}\sum_{i=StartFrame}^{EndFrame}(x_i - \bar{x})^2}$

3. Skewness, $S_{(\det,band,SCA)} = \frac{\left(\frac{1}{nFrames}\sum_{i=StartFrame}^{EndFrame}(x_i - \bar{x})^3\right)}{\sigma^3}$

4. Kurtosis, $K_{(\det,band,SCA)} = \frac{\left(\frac{1}{nFrames}\sum_{i=StartFrame}^{EndFrame}(x_i - \bar{x})^4\right)}{\sigma^4} - 3$

b. Calculate band averages per SCA means ($\bar{X}$) and standard deviations ($\bar{\sigma}$):

i. Initialize data arrays, $\bar{X}, \bar{\sigma} = float(nBands, nSCA)$

ii. For each Band, SCA compute:

1. $\bar{X}_{(Band,SCA)} = \frac{1}{n\det}\sum_{\det=0}^{n\det-1}\bar{x}_{(\det,band,SCA)}$

2. $\bar{\sigma}_{(Band,SCA)} = \frac{1}{n\det}\sum_{\det=0}^{n\det-1}\sigma_{(\det,band,SCA)}$

c. Calculate per band mean and standard deviation.

i. Initialize data arrays, $LampMean = float\ (nBands)$,
$LampStdev = float\ (nBands)$

ii. $$LampMean_{(Band)} = \frac{1}{nSCA} \sum_{fov=0}^{nSCA-1} \overline{X}_{(band,SCA)}$$

iii. $$LampStdev_{(Band)} = \frac{1}{nSCA} \sum_{fov=0}^{nSCA-1} \overline{\sigma}_{(band,SCA)}$$

III. Populate database with the L0c and Level1R statistics

IV. Populate database with selected lamp telemetry

a. Lamp voltages

b. Lamp currents

c. Diode currents

d. Lamp housing temperatures

4.5.11 OLI Lunar Irradiance Characterization

4.5.11.1 Background/Introduction

Lunar acquisitions will be used to complement image quality assessments of the OLI. These will help detect changes in gain, provide a measure of radiometric stability, and reduce absolute radiometric uncertainties.

Changes in gains are determined by comparing the measured lunar irradiances with modeled irradiances from the Robotics Lunar Observatory (ROLO) lunar irradiance model. The IAS derives the measured integrated lunar irradiances, image times, and spacecraft position vectors from the OLI and spacecraft ancillary datasets. The measured irradiances can be compared against modeled lunar irradiances, e.g., those from the USGS Robotics Lunar Observatory, and used to supplement on-orbit calibration trending of the Operational Land Imager.

The L1R lunar product can also be used for further geometrical processing and analysis including creation of geometrically corrected image products (resampled) and MTF characterizations.

Operationally, it is expected that the moon will be imaged once a month (at approximate 7 degrees lunar phase angle) on several Sensor Chip Arrays (SCAs) with one “reference” SCA image always acquired. Information regarding which SCAs are illuminated is also expected from mission operations element since they will be programming the lunar maneuver.

4.5.11.2 Inputs

Description	Symbol	Units	Level	Source	Type
scene_pixel_mean			N_{band} x N_{SCA}	db	Float
Inop Dets			N_{band} x N_{SCA} x N_{det}	CPF	Int

Description	Symbol	Units	Level	Source	Type
Scene (L1r)		$W/m^2/sr/\mu m$	N_{band} x N_{SCA} x N_{det}		Float
Odd/even detector offsets			N_{band} x N_{SCA} x N_{det}	CPF	
ImpNoise			N_{band} x N_{SCA} x N_{det}	LM	Int
Sat Pixels			N_{band} x N_{SCA} x N_{det}	LM	Int
Dropped Frames			N_{band} x N_{SCA}	LM	Int
Median Filter Size	β			CPF	Int
Integration Threshold Factor	δ		N_{band}	CPF	Float
Irradiance Conversion			N_{band}	CPF	Float
Along IFOV			N_{band}	CPF	Float
Data/Imagery start/stop time [UTC]	T_{start}, T_{end}	YYYYDDDh hmmss.sss		Ancillary/db	Text or Date
Frame rate	F	Hz		Ancillary/db	Float
Spacecraft ephemeris/Attitude		Km		Ancillary/db	Float

4.5.11.3 Outputs

Description	Symbol	Units	Level	Target	Type
Integrated Irradiances	SumIrr	$\mu W/m^2/nm$	N_{band} x N_{SCA}	Db and text file	Float
Apparent lunar YSize [mrad and frames]	Ysize	Mrad, frames	N_{band} x N_{SCA}	Db and text file	Float
Frame numbers at moon center and top and bottom of lunar edges.		frames	N_{band} x N_{SCA}	Db and text file	Int
S/C attitude and attitude rates at top, middle and bottom of lunar image		rads, rads/s or arcsec, arcsec/s [as provided in the ancillary db]	N_{band} x N_{SCA}	DB and text file	Float
Time at center of lunar image [UTC]	t_m	YYYYDDDh hmmss.sss	N_{band} x N_{SCA}	DB and text file	Text or Date
SC PositionVector [J2000 GCI] at center of lunar image [UTC]		Km		Db and text file	Float
Moon-SC, Earth-Sun, Earth-Moon, and Earth-SC		km		Db and text file	Float
Lunar Mask		boolean	N_{band} x N_{SCA} x N_{det}	Image File	Int

4.5.11.4 Options

1. Output lunar mask for verification.
2. Text output of Integrated Irradiance, apparent lunar y-sizes, start and stop frame numbers of lunar images, and location of the middle of the lunar images for each band and in every illuminated SCA.

4.5.11.5 Procedure

1. Check the scene_pixel_mean of each SCA to determine the brightest one, which should contain the image of the moon. This can be done per band, or it can be done once since the moon will be in the same SCA for all bands.
2. If the illuminated SCA contains an inoperable detector, report an error and exit.
3. For each illuminated SCA and Band calculate inputs to Rolo
    a. Read image data, $L1R_{iband,isca}$
    b. If the input L1R data implemented any odd even offsets, then remove odd/even detector offsets from $L1R_{iband,isca} \rightarrow L1C_{iband,isca}$. Otherwise, $L1C_{iband,isca} = L1R_{iband,isca}$
    c. Create Lunar Mask, $LunM_{iband,isca}$:
        i. Skip ImpNoise, Sat Pixels, and Dropped Frames
        ii. Remove artifacts (bright stars, etc.) from image: Filter each column with median filter, size = β. β is read from the CPF: Median Filter Size (default=5).
        iii. Obtain maximum radiance value, $MaxRad_{iband,isca} = \max(L1C_{iband,isca})$
        iv. Set irradiance Threshold = $MaxRad_{iband,isca} * \delta$, where δ is the Integration Threshold Factor from the CPF (default = 0.8)
        v. LunM = 1 where $L1C_{iband,isca} \geq$ Threshold, else LunM = 0
    d. Calculate Irradiance, $SumIrr_{iband,isca}$
        i. Sum radiance over lunar images, $SumRad_{iband,isca} = \sum LRad*LunM$
        ii. Convert $SumRAD_{iband,isca}$ to Irradiance in $\mu W/(m^2.nm)$,
        iii. $SumIrr_{isca}=SumRAD_{isca}$*Irradiance Conversion (default for MS = 1.81077e-5 for PAN = 4.52694e-6
    e. Calculate apparent lunar y-size:
        i. For each detector locate start/end frame number where LunM = 1, $start_{idet}$, end_{idet}
        ii. Number of mask pixels in each column: $N_{idet}=end_{idet}-start_{idet}$
        iii. $Ysize_{iband,isca} = Maximum(N_{idet})$, at column location $Dmax_{iband,isca}$. I.e. Y-size is the maximum number of pixels set to 1 in the lunar mask and Dmax is the detector number where that maximum occurs.
        iv. Ysize in mrads,: $mYsize_{isca} = Ysize_{isca} * \Omega$, where Ω is the detector solid angle or Along track IFOV (default for MS = 4.2435e-5 for PAN = 2.1217e-5)
    f. Calculate UTC time at center of moon, UTC_Moon(Band,SCA):
        i. Center of moon in image, FrameNumberMoonCenter = (start(Dmax)+end(Dmax)/)/2
        ii. UTC_moon=(Image start time)+FrameNumberMoonCenter/(Frame rate)
    g. Read spacecraft ephemeris at UTC_moon: J2000 Position Vector(X,Y,Z).
    h. Calculate Moon-SC, Earth-Sun, Earth-Moon, and Earth-SC distances with NOVAS library.
4. Outputs

a. Obtain and populate database with integrated irradiance results and ancillary data. These include the following:
    i. Lunar Integrated Irradiance,
    ii. Apparent lunar Y-sizes (in mrads and frame count),
    iii. Frame numbers at moon center and top and bottom of lunar edges,
    iv. S/C attitude and attitude rates at top, middle, and bottom of lunar image,UTC time at center of lunar image,
    v. Spacecraft position vectors at center of lunar image,
    vi. Moon-SC, Earth-Sun, Earth-Moon, and Earth-SC ranges,
    vii. Lunar Mask.

4.5.12 OLI Reflectance Conversion

4.5.12.1 Background/Introduction

The standard Level 1T product will be a top of atmosphere reflectance product. This algorithm will convert the radiance image to a reflectance image in a per-scene operation. The two products are linearly related to each other by a band specific coefficient that is proportional to the exoatmospheric solar irradiance in each band and the Earth-Sun distance for the scene's day of acquisition. The per-band coefficients will be determined once on orbit, after the first look at the diffuser. For prelaunch testing, an estimate of the coefficient can be derived from the exoatmospheric solar irradiance. The reflectance values will be between 0.0 and 1.0.

Since all problem pixels should have been corrected by this point in the processing flow, this algorithm assumes that every image pixel is a valid radiance value. Thus there is no consideration for dropped frames, inoperable detectors or saturated pixels.

This algorithm will only process OLI data, not TIRS data. The equivalent algorithm for TIRS data is Temperature Conversion.

4.5.12.2 Inputs

Description	Symbol	Units	Level	Source	Type
Radiance image data	L	W/m^2 sr um	N_{band} x N_{pixel}		float
Earth-Sun Distance	d	AU	scalar	JPL ephemeris table	float
Radiance to Reflectance Conversion Coefficients	R_ρ	sr/(w/m^2 um)	N_{band}	CPF	float

4.5.12.3 Outputs

Description	Symbol	Units	Level	Target	Type
Reflectance image	ρ	[]	N_{band} x N_{pixel}		float
Reflectance to radiance conversion coefficients	ρ_R	(W/m^2 um)/sr	N_{band}	Radiance Rescaling	float

4.5.12.4 Procedure

1. For each band, apply conversion coefficients ($R\rho$) to radiance images (L1r). This equation converts the image radiance [W/m^2 sr um] to reflectance [unitless].

   a. $$\rho = d^2 \cdot L \cdot R_\rho \quad (1)$$

   b. where L is the image radiance and d is the Earth-sun distance (AU) for the day the scene was acquired.

2. For each band, calculate reflectance-to-radiance conversion coefficient (ρR).

   a. $$\rho_R = \frac{1}{d^2 \cdot R_\rho} \quad (2)$$

   b. This coefficient is passed to Radiance Rescaling.

4.6 TIRS Radiometry Algorithms

4.6.1 TIRS Bias Model Calibration

4.6.1.1 Background/Introduction

Conversion from instrument digital counts (DN) to radiance (W/m^2-sr-μm) occurs in 3 steps: response linearization, bias removal, and gain application. The bias that is removed in the second step is a combination of the dark and background response of the instrument, and is the total response of the instrument to “nothing,” or a very cold target. On orbit the TIRS instrument will collect data while looking at deep space. The per-detector means of the data from these collects can be used as an estimate of the cumulative dark and background responses of the instrument.

This algorithm only needs to be implemented as a part of Ingest, and should not be run using long (meaning longer than the typical 2 seconds) space look collects.

4.6.1.2 Inputs

Description	Symbol	Units	Level	Source	Type
Per Detector Means from latest space look (TIRS)	S_{TIRS}	DN	N_{band} x N_{SCA} x N_{det}	Histogram Statistics	Float

4.6.1.3 Outputs

Description	Symbol	Units	Level	Target	Type
Per Detector Means from latest space look (TIRS)	S_{TIRS}	DN	N_{band} x N_{SCA} x N_{det}	BPF	Float

4.6.1.4 Options

- Start and stop date/time of desired means (T0 and T1). Normally T0 should be the stop of the pre-acquisition deep space collect and T1 should be the start of the post-acquisition deep space collect.

4.6.1.5 Procedure

Retrieve the histogram statistics from one collect each of TIRS space look data acquired prior to the desired start date/time (T0) and after the desired stop date/time (T1) and write these to the BPF database.

4.6.2 TIRS Dark Response Determination

4.6.2.1 Background/Introduction

One component of the signal recorded by each TIRS detector is the result of the response to the thermal energy in the QWIP material, called the Dark Response. This value varies by detector for all bands and is dependent on detector temperature, bias voltage, integration time and detector gain. Nominal values are determined during ground testing at the component level by using a cold shutter to mask practically all photon flux from the detectors. In-flight, the two Earth-imaging “illuminated” bands cannot be shuttered from the photons emitted by the instrument structure and optics, thus their dark response cannot be directly measured. A third TIRS band, the “dark” band, is permanently shuttered. The response for this dark band is continuously measured and available along with the illuminated band data. If there is variation in the dark response on orbit, e.g., as due to detector temperature variation, this dark band will capture it.

This algorithm analyzes the response from the dark band, compares it to the dark response from ground testing and calculates an estimate of the change in dark response. This estimate can be extrapolated to a change in the dark response of the illuminated band data and applied to the radiometric calibration in later steps. The application of the correction is dependent on the evaluation of how well the dark response of the dark band correlates to the dark response of the illuminated bands. Since the dark response of a QWIP detector is a function of its individual gain, the difference in gains between the dark and illuminated detectors must be accounted for in the evaluation.

This algorithm can be run on sets of calibration observations as a stand-alone algorithm to evaluate the dark response correlation and stability. It can also be run on a scene-by-scene basis as part of the primary radiometric calibration. In the former case, CPF values can be evaluated and adjusted to reflect the best in-flight knowledge of the dark response. This process will also be necessary if the set points for integration time, bias voltage, detector temperature, or A/D offset level are changed in flight. In the latter case, this algorithm is needed in-line for product generation.

The TIRS instrument is expected to be stable in terms of radiometric gain and bias such that calibration updates should not be required on an orbital or daily basis. In this scenario the dark response is treated as part of the overall background response and this algorithm may not substantially improve the precision of the radiometric calibration. In the event that the focal plane temperature shows low frequency variation, lower than per scene, this correction may improve the absolute radiometric calibration precision.

4.6.2.2 Inputs

Description	Symbol	Units	Level	Source	Type
Scene Mean (dark band only)	$\bar{Q}$	DN	N_{SCA} x N_{det}	DB (Histogram Statistics Characterization)	Float
Baseline Dark Response	D_B	DN	N_{band} x N_{SCA} x N_{det}	CPF	Float
Detector Relative Gains	G_{Det}	Unitless	N_{band} x N_{SCA} x N_{det}	CPF	Float
SCA Relative Gains	G_{SCA}	unitless	N_{band} x N_{SCA}	CPF	Float
Absolute Gains	G_{Abs}	$\frac{DN}{W/m^2 sr\ \mu m}$	N_{band}	CPF	Float
Reference Absolute Gains	G_{Ref}		N_{band}	CPF	Float

4.6.2.3 Outputs

Description	Symbol	Units	Level	Target	Type
Dark Response	D	DN	N_{band} x N_{SCA} x N_{det}	Test CPF or Bias Removal	Float

4.6.2.4 Options

- Output adjusted dark response to file (default off). This is mainly for the standalone implementation of this algorithm, but could also be helpful for anomaly resolution.
- Output baseline dark response or adjusted dark response (default is baseline dark response from the CPF).
- Use reference absolute gains instead of the 'normal' absolute gains

4.6.2.5 Procedure

1. Read the baseline dark response (D_B) for each band/SCA/detector from the CPF (including the dark band)
2. If baseline dark response option set, output the dark response (D_B) and exit, else continue
3. Read the gain coefficients (G_{Aba} *or optionally* G_{Ref}, G_{SCA}, and G_{Det}) for each band/SCA/detector from the CPF (including the dark band) and calculate the per-detector absolute gain (G)

$$G(b,s,d) = G_{Abs}(b) G_{SCA}(b,s) G_{Det}(b,s,d) \quad (1)$$

4. Read the scene average response ($\bar{Q}$) for each SCA/detector for the dark band from the histogram statistics for the input scene or calibration collection being processed.
5. For each SCA:
   5.1. For each detector in the Dark band:
      5.1.1. Calculate the difference between the baseline dark response (D_B) and the scene average dark response ($\bar{Q}$), giving the dark response difference (Δ_D)

$$\Delta_D(s,d) = D_B(s,d) - \bar{Q}(s,d) \qquad (2)$$

5.1.2. Divide the dark response difference (Δ_D) by the detector gain (G), giving the adjusted dark response difference (Δ_A)

$$\Delta_A(s,d) = \frac{\Delta_D(s,d)}{G(s,d)} \qquad (3)$$

5.1.3. Calculate the average adjusted dark response difference (Δ_S) across all operable dark detectors in the SCA, $\Delta_S(s) = \frac{1}{N_d}\sum_{d=1}^{N_d}\Delta_A(s,d)$ (4)

5.2. For each Band/Detector:

5.2.1. Multiply the SCA average adjusted dark response difference by each detector's gain and add the result to the baseline dark response for each detector, giving the adjusted dark response for the detector.

$$D_A(b,s,d) = G(b,s,d)\Delta_S(s) + D_B(b,s,d) \qquad (5)$$

Downstream algorithms can use either the baseline dark response (D_B) or the adjusted dark response (D_A), in either case it is represented as simply the dark response (D).

4.6.3 TIRS Bias Removal

4.6.3.1 Background/Introduction

Conversion from instrument digital counts (DN) to radiance (W/m^2-sr-μm) occurs in 3 steps: response linearization, bias removal, and gain application. The second step (bias removal) is described in this document. Although the name implies this algorithm just removes the detector bias, it actually removes the dark response and background response (which can come from the dark and background response determination algorithms or from per detector means of deep space look collects), as well as an offset that is part of the gain function that converts linearized, background subtracted DN to radiance. Note that the order of these three steps is different than OLI because the "bias" being removed is a function of detector voltage and temperature and must be linearized before being subtracted.

Bias removal is accomplished by subtracting a value (in linearized DN) from each pixel of the input image. This value varies by detector for all bands. The values are determined initially during ground testing and made available via the CPF. In-flight calibration observations (space-look data) can be used to verify the per-detector background signal and adjust the CPF periodically if necessary.

4.6.3.2 Inputs

Description	Symbol	Units	Level	Source	Type
Scene (Linearized DN)	Q_L	DN	N_{band} x N_{SCA} x N_{det} x N_{frame}	Response Linearization	Float
Pre-acquisition Deep Space Averages	S_a	DN	N_{band} x N_{SCA} x N_{det}	BPF	Float
Post-acquisition Deep Space Averages	S_b	DN	N_{band} x N_{SCA} x N_{det}	BPF	Float
Dark Response	D	DN	N_{band} x N_{SCA} x N_{det}	Dark Response Determination or CPF	Float
Background Response	B	DN	N_{band} x N_{SCA} x N_{det}	CPF	Float
Gain Function Offset	Go	DN	N_{band} x N_{SCA} x N_{det}	CPF	Float

4.6.3.3 Outputs

Description	Symbol	Units	Level	Target	Type
Scene (Bias Subtracted, Linearized DN)	Q_{LB}	DN	N_{band} x N_{SCA} x N_{det} x N_{frame}	TIRS Second Linearization	Float

4.6.3.4 Options

- Dark and Background Response Selection

    1. Pre-acquisition deep space averages (S_a)
    2. Post-acquisition deep space averages (S_b)
    3. Average of pre-and post-acquisition deep space averages (S_{ab}) (default)
    4. dark and background responses from the CPF (or dark response determination in the case of dark response)

4.6.3.5 Procedure

1. If pre- or post-acquisition deep space averages have been selected, then retrieve the selected deep space averages. Whichever is selected will be referred to as *S(b,s,d)* for the remainder of the algorithm, where *d*=detector, *s*=SCA, and *b*=band.

   If the average of the pre-and post-acquisitions is selected, then retrieve the selected deep space averages and calculate S using equation (1).

$$S(b,s,d) = \frac{1}{2}\left(S_A(b,s,d) + S_B(b,s,d)\right) \quad (1)$$

   If the dark and background responses from the CPF are desired, then retrieve the dark and background responses from the CPF (or dark response determination in the case of dark response) and calculate *S* using equation (2).

$$S(b,s,d) = D(b,s,d) + B(b,s,d) \quad (2)$$

2. To calculate the total bias, add the gain function offsets to the combined dark and background responses.

$$bias(b,s,d) = S(b,s,d) + G_O(b,s,d) \quad (3)$$

3. Remove the bias from the linearized scene.

$$Q_{LB}(b,s,d,f) = Q_L(b,s,d,f) - bias(b,s,d) \quad (4)$$

4.6.4 TIRS Non-linear Response Characterization

4.6.4.1 Background/Introduction

The output of the TIRS instrument is quantized output (Q) in units of digital number (DN). This Q is expected to be related to the input signal of the detectors, but that relationship may not be linear. Each detector may have unique non-linear irregularities in response that must be corrected in processing.

Figure 4-102 shows a response slope for a typical detector from Band 1, SCA 1. All detectors studied exhibit similar behavior.

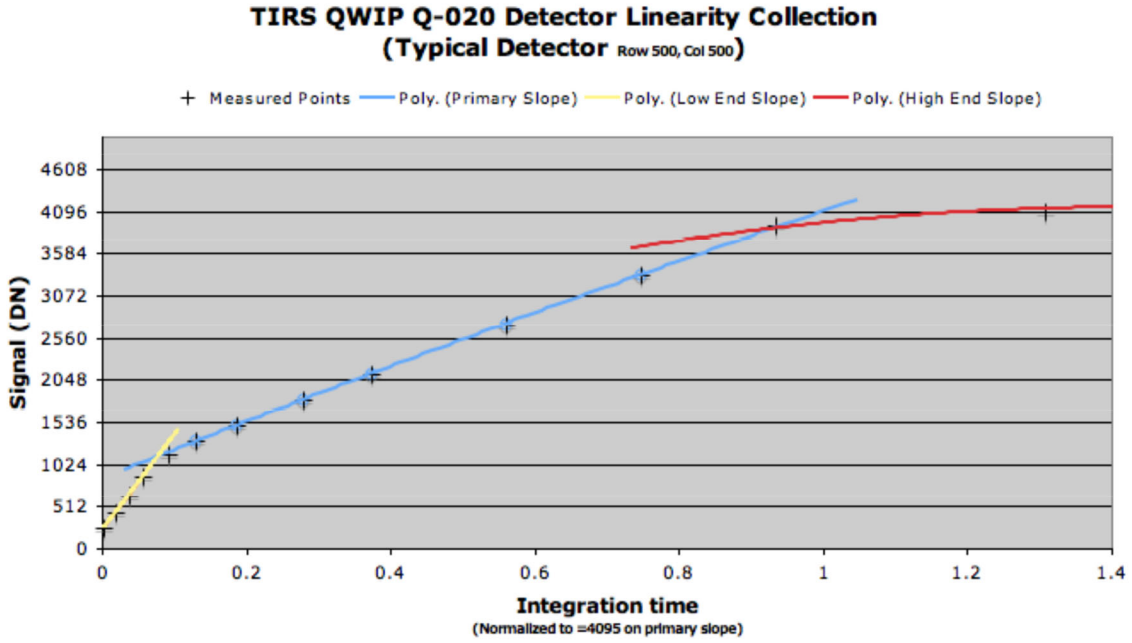

Figure 4-102. Typical QWIP Detector Response Shape

From test collections made either prelaunch or in orbit, a set of parameters can be derived to linearize the detector response. The preferred data should be Integration Time Sweep (ITS) collections made with the internal blackbody panel as a background.

The intended form of the linearization equation is piecewise quadratic, with three distinct regions. The cutoff points between the regions – the points where the functions intersect – are determined by equating the adjacent functions. The first (bottom) region extends from zero up to the first minimum of the linear fit residual plot. The second (middle) region extends up to the last experimental point within the detector's dynamic range.

The third (top) region covers the most non-linear portion of the detector response – from the top of the detector's dynamic range to the high analog saturation point.

4.6.4.2 Inputs

Description	Symbol	Units	Level	Source	Type
L0 (Bias corrected) Mean for each integration time collection	Q	DN	N_{band} x N_{det} x N_{level}	Db (Histogram)	Float
Radiance level for each integration time collection	R	W/m^2 sr µm	N_{level}	Db (Histogram)	Float
Integration time for each collection	i		N_{level}		Float

4.6.4.3 Outputs

Description	Symbol	Units	Level	Target	Type
Remapping function cutoff thresholds		DN	N_{band} x N_{SCA} x N_{det} x N_{cutoff}	CPF	Float
Remapping function parameters			N_{band} x N_{SCA} x N_{det} x N_{coeff} x ($N_{cutoff+1}$)	CPF	Float
Mean absolute residual		DN	N_{band} x N_{SCA} x N_{det}	Report file	Float
Maximum residual		DN	N_{band} x N_{SCA} x N_{det}	Report file	Float

Note: N_{cutoff} equals 2 in the current implementation.

4.6.4.4 Options

The cutoff point table below (in Procedure Section 2.1) contains an array of floating point values that are input to the work order but that should be editable by the operator.

4.6.4.5 Procedure

1.0 Prepare Data

Each integration time sweep collection for TIRS will involve a blackbody collection taken at an integration time setting, which is given in the metadata. For each collection the mean DN value for each detector should be stored in the database by the Histogram Statistics procedure.

To prepare data for linearity characterization, the collections to use should be identified and the per-detector means obtained. For each collection the solar angle and viewed radiance for each band should be calculated. Once collated, the prepared data should be an array of [Q,R,i] ([mean DN, viewed radiance, integration time]) for every band and detector.

2.0 Linearity Characterization

2.1 Find preliminary cutoff points.

The cutoffs are initially chosen as the available effective radiances that are nearest to the radiances from Table 4-63 below:

Band	Band name	First cutoff R (W/m^2 sr µm)	Second cutoff R (W/m^2 sr µm)
10	10 µm band	9.64	20.5
11	12.8 µm band	6.0	15.0
12	alternate 10 µm band	9.64	20.5
13	alternate 12.8 µm band	6.0	15.0

Table 4-63. TIRS Linearity Cutoff Radiances

The effective radiance of each integration time sweep collection is calculated:

$R(i) = R_{max}*i$

The collection whose effective radiance is nearest to the first cutoff point is labeled i_{low} :

$Q_{low} = Q(d, i_{low})$
$R_{low} = R(i_{low})$

The collection with effective radiance nearest to the second cutoff point is labeled Q_{high} (with corresponding R_{high} and i_{high}).

These labels are made once for each band. These 'a priori' radiance cutoff levels should be work order parameters that are adjustable by the IAS analysts.

2.2 Identify saturation level

For each detector, if the maximum DN value is not at the highest integration time, then the detector saturates at some point that can be specified as Q_{sat}, with a corresponding i_{sat}.

The value of i_{sat} should be stored and used as an effective end to the input data. No saturated data should be used to derive the quadratic fits.

The preliminary cutoff point Q_{high} should be adjusted so that it is less than the saturation level, with enough points in between to allow a fit to be made. Because ITS collections are discrete steps, this means that the value of i_{high} should be set to i_{sat}-2, or two steps less than the effective DN value of isat.

if $Q_{high} > Q_{sat}$, then
$i_{high} = i_{sat} – 2$
$Q_{high} = Q(d, i_{sat} – 2)$

If this brings the value of Q_{high} to less than the value of Q_{low}, then Q_{low} must be decreased in a similar fashion:

if $Q_{low} > Q_{high}$, then
$i_{low} = i_{high} - 2$
$Q_{low} = Q(d, i_{high} - 2)$

While saturation is common, it should be a rare occurrence for the saturation level to be less than Q_{high}.

2.3 Create idea line

Once the cutoff points have been found, the ideal line for each detector can be calculated. On TIRS, linear behavior is assumed to pass through the origin (0 DN, 0 Radiance), and it is normalized to the lowest recorded integration time (i_1). The slope of the ideal line is then:

$$M(d) = Q(d, i_1) / R(i_1) = Q(d, i_1)/(R_{max}*i)$$

An ideal line curve is then created, with one point for every integration time:

$$Q_{ideal}(d,i) = M(d)*R_{max}*i$$

2.4 Polynomial fit

The parameters of the linearization are then found by fitting the actual data to the ideal line. This is done in three regions: 0 to i_{low}, i_{low} to i_{high}, and i_{high} to i_{sat}.

The fit is currently done with the IDL routine POLY_FIT, which uses the matrix inversion method that can be found in Numerical Recipes. Any implementation of a two-dimensional polynomial fit can be used. The output of each fit for each region should be quadratic coefficients such that:

$$Q'(d,r,i) = c(r,2)*Q(d,r,i)^2 + c(r,1)*Q(d,r,i) + c(r,0) \approx Q_{ideal}(d,r,i)$$

where $Q'(d,r,i)$ = The linearized DN values (one for every i).
$c(r,n)$ = The *nth* order quadratic coefficient for region *r*.

2.5 Determine actual cutoff points.

The actual cutoff points should be calculated after the quadratic coefficients, so that the fit between regions is as smooth as possible. The

cutoff points are calculated by finding the point at which the fitted curves for two regions meet. This point can be calculated with the quadratic equation:

$$Q'_{low} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

where a = The difference in the second-order coefficients for regions 1 and 2.
a = c(2,2) – c(1,2)
b = The difference in the first-order coefficients for regions 1 and 2.
b = c(2,1) – c(1,1)
c = The difference in the zeroth-order coefficients for regions 1 and 2.
a = c(2,0) – c(1,0)

The quadratic equation has two solutions, due to the ± in the equation, and neither solution is guaranteed to be a real or positive number. Both solutions should be calculated, and the real component compared to the 'a priori' cutoff point. The solution that is closest to the first 'a priori' cutoff value Q_{low} should be chosen as the final cutoff value.

Similarly, Q'_{high} is calculated between the second and third regions (c(3,n) – c(2,n)), and its two solutions are compared to the second 'a priori' cutoff, Q_{high}.

2.6 Report output.

After the parameters have been created, the data should be linearized using them and residuals to the ideal line should be calculated. This array of residuals is used by the IAS analyst to check the accuracy of the linear characterization.

$Residuals(d,i) = Q'(d,i) – Q_{ideal}(d,i)$

This residual array is only calculated up to i_{sat}. The residuals to the ideal line in the saturated detector region should not be reported.

The mean and maximum absolute residual for each detector should be reported in an output file.

4.6.5 TIRS Response Linearization

4.6.5.1 Background/Introduction

The TIRS readout electronics produce a quantized output (Q) in units of digital number (DN) representing the amount of signal measured by each detector. This Q is expected

to be related to the input signal in a non-linear way. The non-linearity is different for each detector. To simplify parts of radiometric correction and calibration, the Q must first be adjusted to compensate for this non-linearity.

Response linearization data are collected using multiple integrations times with a stable signal source. During ground calibration the stable source is a shuttered dewar, a uniform calibration source, or the TIRS Calibration GSE. On-orbit, the stable signal source can be either the on-board calibrator or the space view. Plotting the Q vs. Integration Time illustrates the non-linearity in the detector response as the electron well fills. This methodology assumes that linearity with integration time is equivalent to linearity with radiance.

The 9803 Readout Multiplexers used in the TIRS focal plane have two distinct slopes. The initial response of output signal versus input signal has a relatively high slope for approximately the first quarter of the detector saturation level (low-end slope). The majority of the dynamic range of the detector/readout system (primary slope) has a response about 1/3 of the low-end slope. Nearing analog saturation the response slope flattens out (high-end slope). Additionally there are non-linearities in each response slope due to the ROIC output buffer electronics, and the analog to digital circuitry.

Figure 4-103 shows the response slopes for a typical detector/readout system from one of the TIRS SCAs. For convenience the Integration Time is normalized to the point where the primary slope output signal would be 4095 DN.

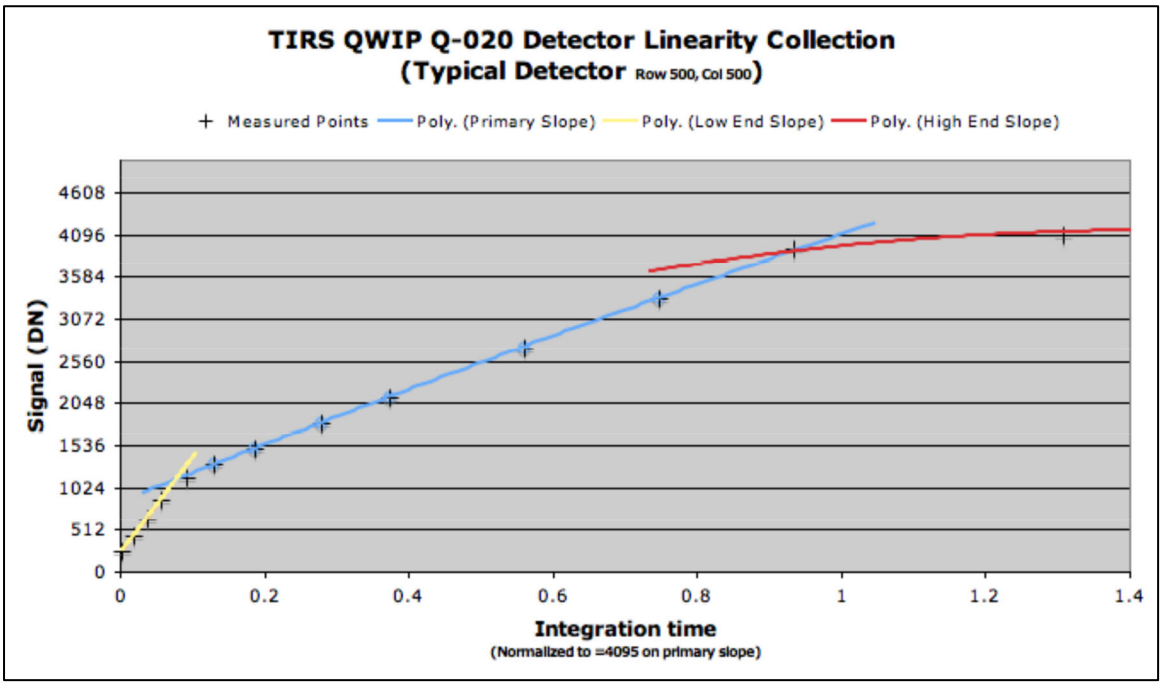

Figure 4-103. Typical QWIP Detector Response Shape

Figure 4-104 shows the inverted response plot with each response slope fit to a quadratic function. This data is derived from the Detector Linearity Collection detector level data. There are not enough data points at this time to derive a quadratic function for the low-end slope, so the square term is set to zero. The high-end slope also lacks enough data points at this time for a good fit. The high-end slope is also likely to fit an exponential function better than a quadratic. The theoretical form of the curve is not known at this time, but the piecewise quadratic/linear approximation has been determined to be valid.

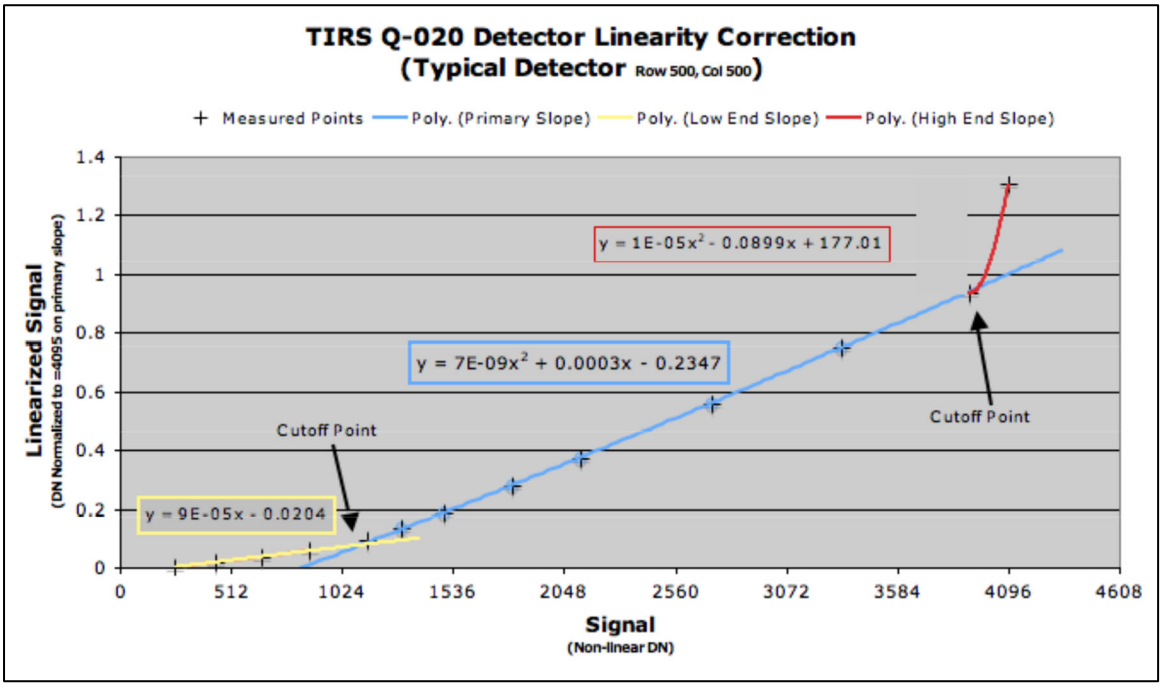

Figure 4-104. Inverse Detector Response with Quadratic Fits

The instrument is designed so that the scene and OBC data will be in the primary slope portion of the response curve. Depending on the focal plane temperature, integration time, bias voltage, and other factors, the dark band data and deep space looks may return levels in the range of the low-end slope. In order to correct the scene data using the dark response it is necessary to match the response of the low-end slope to the primary slope. The cutoff point where the two slopes intersect is determined and the appropriate linearization correction is made. The instrument will likely be tuned so the analog saturation point associated with the high-end slope is above the digital saturation point, and thus these data will never be seen. Processing the high-end slope is included in case it is needed.

The mapping function used in this algorithm is a quadratic for all three slopes. The cutoff points are determined by equating the two adjacent quadratics functions. Analysis has shown that a higher order polynomial does not add any benefit to the linearization to the low end slope or primary slope of the response curve

4.6.5.2 Inputs

Description	Level	Source	Type
Scene (L0R)	N_{band} x N_{SCA} x N_{det} x N_{frame}		Float
Remapping function cutoff thresholds	N_{band} x N_{SCA} x N_{det} x N_{cutoff}	CPF	Float
Remapping function parameters	N_{band} x N_{SCA} x N_{det} x N_{coeff} x ($N_{cutoff+1}$)	CPF	Float

Note: The planned implementation is for two cutoff points (N_{cutoff} = 2) and three sets of quadratic coefficients (N_{coeff} = 3).

4.6.5.3 Outputs

Description	Level	Target	Type
Scene (linearized)	N_{det} x N_{frame}		Float

4.6.5.4 Procedure

Response linearization should be a simple replacement of the incoming value with a linearized response value.

The linearized value will be calculated for each sample point by an algorithm. These algorithms will be determined by prelaunch calibration of the detectors. As described above, this algorithm is a multi-segment quadratic function with the detector Q value as its only variable. Parameters for the linearization function will be stored in the CPF. For this method the required parameters are three sets of quadratic coefficients (one set for each sloped segment), and two cutoff points.

1.1. For each input frame
1.2. For each detector
  1.2.1. Compare input Q with cutoff values and select appropriate quadratic coefficients.

    The input Q value is compared to the cutoff thresholds in the CPF, to determine which segment of the linearization to use.
    If input is less than cutoff #1, then segment 1 is used.
    If input is greater than or equal to cutoff #1 but less than cutoff #2, then segment 2 is used.
    If input is greater than or equal to cutoff #2, then segment 3 is used.

  1.2.2. Evaluate the quadratic function at the input Q:

    $output = qp[0, s] + qp[1, s] * input + qp[2, s] * (input)^2$

    where input = the input value, Q, in DN.
    output = the output value, Q_1, in linearized DN.
    qp[x, s] = linearization parameter x, for segment s.
    These parameters come from the CPF.

1.2.3. Return the output value.

4.6.6 TIRS 40 Minutes Radiometric Stability Characterization

4.6.6.1 Background/Introduction

This algorithm provides for on-orbit characterization of the radiometric stability of the TIRS, specifically related to TIRS RD requirement (TIRS-547) “Thermal band data for all pixels, after radiometric calibration per 5.3.1.2, for radiometrically constant targets with radiances greater than or equal to the radiance corresponding to TTypical, shall not vary by more than plus or minus 0.7% (1-sigma) of their radiance over a 40 minute period.” This algorithm is to be implemented, at least initially, as part of the Calibration/Validation Toolkit (CVTK).

For the TIRS apparent response stability is related to multiple contributing factors:

1. Background response and dark response stabilities over the 40 minutes
2. The reference constant source stability stabilities over the 40 minutes
3. Validity of CPF used to produce the radiometric product such parameters include the non-linearity correction and the conversion to radiance parameters.
   In other words, if any of these CPF parameters may have some thermal or temporal dependence that are not accounted for, it would impact the interpretation of the actual response stability results.

Furthermore, although primarily focused on product stability this tool can also be used to understand the stabilities of several of the instrument sub-systems. Establishing baseline characteristics of the instrument stability will help in verifying the time it will take to return to nominal operation after lunar collects or other non-routine orbital operations. Baseline time duration for such return-to-nominal period would be based on telemetry information and could be complimented by the metric generated by this tool.

Key telemetry information associated with TIRS stability includes: Temperatures, (OBC, Telescope, SSM, FPA, FPE, MEB); Current and Voltages (OBC and FPE).

This algorithm uses deep space and OBC TIRS data to assess the within orbit product and instrument stability.

The basic concept of this algorithm is to analyze TIRS statistics from multiple 60sec segments within the stability evaluation interval of 36 min or 1.5 orbit. For each of the 60sec segments it computes the instantaneous stability parameters, which can be used then to gain understating on stability changes over the entire evaluation interval. Computing the 60sec stability parameters the algorithm is based on Hist_stat processing outputs.

Information about Input data types expected

Two types of data collects will be used for these analyses: on board calibrator data and deep space data. The OBC data will be at or above the temperature of 290K (current plan is to use 290K, 295K and 315K) then, manual interpolation analysis will be required

to transform the stability results into the requirement temperature of $T_{typical}$. Currently during on-orbit commissioning the main data collects that will be used by this algorithm are the continuous 36min data of OBC or deep space data as well as a repeated sequence of 30 Cal-sequence files over 1.5 orbit with once every 5 minutes collect sequence of 1 min deep space and 1 min OBC will be available.

This algorithm is designed primarily to work on TIRS data but could be converted to be applicable for both TIRS and OLI 36 minutes long collect. This algorithm can operate on all 3 TIRS bands (10µm,12µm, blind) but practical use will need only the 10µm and 12 µm bands data for product level performance assessment. For that reason the analyst can use a selection switch to select if this algorithm should process stability of TIRS blind data or just the imaging bands.

Similar to the 60sec stability algorithm this algorithm would rely on Histogram statistics Characterization processing to collect the basic statistics from either multiple scenes given in a sequence to form a virtual long interval or sub-segments of a long interval (that may be split into multiple files). The algorithm will populate the minimal set of radiometric stability metric as a standalone algorithm that is gathering the instantaneous stability information on each of the scenes or segments in the interval. In the case the scene provided is one long 36 min dataset (composed of several mission files) the algorithm will pre-process the input data to produce multi-segments statistics information needed for processing using a moving window of 60sec. The interpretation of the results requires that the scenes are temporally constant radiance, e.g., deep space and OBC acquisitions (those could be part of a special Cal sequence sweep with enough dwell time for the collections of the number of lines needed).

For deep space collect we use (L0r) input that will only be non-linearity corrected. For OBC collects we use both (L0rc) and (L1r) products as inputs.

Information about Output produced and its interpretation

The output of this tool enables trending of: instantaneous stability of segments for the response, and stability over the 36min duration or longer.

The output mimics the 60s stability algorithm configuration where the output depends on the input data provided, i.e., OBC or deep space. Stability statistics across the scene segments are calculated from these uniform radiance scenes and stored in the characterization database. Using collects made at the nominal integration time the statistics are calculated by Histogram Statistics. Multiple levels of processing will be used by this algorithm. Six categories of outputs will be produced:

1. Background response band average stability (deep space, non-linearity corrected)
2. OBC response band average stability – (background corrected and non-linearity corrected)
3. Radiometrically corrected OBC response stability in radiance
4. Radiometrically corrected OBC response stability in %

5. Optional – Detector-by-detector Radiometrically corrected OBC response stability in % (for operability characterization)
6. Optional – Detector by detector background response stability (for operability characterization)

The OBC L1r product stability metric will be converted from radiance units to % so it can be evaluated against the 0.7% requirement level. OBC L0rc data will be used to characterize the band mean net response stability in the various segments over the data duration interval (36 min or longer).

Additional considerations
It is recommended stability acquisitions be routinely collected and assessed, in the following conditions: after standard TIRS operations in either Earth sun-lit or night part of the orbit, after Lunar or Solar collects with 5 minutes margin from the time TIRS returned to Earth view nadir pointing (the 5min period is driven by TIRS requirements and SC pointing stability allocated period).

The output of this algorithm can be used by subsequent algorithms that trend detector-level characteristics.

TIRS radiometric conversion parameters are assumed invariant after the background corrected response is computed. Trending of the output of this stability algorithm could assist in validating this assumption.

Future developments on this characterization processing may include the use of OLI data and TIRS Earth - night ocean long passes.

To obtain a better representation of the TIRS stability will require additional analysis and correction of any influences of low frequency drifts in the TIRS and OBC temperatures induced by the data collection conditions. This will require the analysis to correlate between the trended results to the various subsystems telemetry information.

4.6.6.2 Inputs
This algorithm works with two types of source data: a single long continuous collect or multiple short segments over a longer period.

The expected input will be produced by Hist_stat or the special long data Hist_stat algorithms which would operate on either a pre-processed data given in the format of a set of 30 4200 lines files (covering the period of 150 minutes) illustrating 30 interval sub-segments, or pre-processed data given in the format of a set of 36 4200 lines files illustrating 36 interval sub-segments (covering the continuous 36 min collect).

Inputs per segment are highlighted; inputs common to the full dataset are not highlighted. Note that when correlating telemetry data to this analysis it should be done in the same timing intervals of the segments used for the instrument response.

Description	Symbol	Units	Level	Source	Type
Interval Segment Signal Mean	$\overline{Q}_{B,S,D}$	DN or W/m^2 sr µm	N_{band} x N_{SCA} x N_{det}	Long collect statistic Characterization Stat Char $\overline{Q}$	Float
Interval Segment Signal Max	$Max\_Q_{(B,S,D)}$	DN or W/m^2 sr µm	N_{band} x N_{SCA} x N_{det}	Long collect statistic Characterization Stat Char Q_{max}	Float
Interval Segment Signal Min	$Min\_Q_{(B,S,D)}$	DN or W/m^2 sr µm	N_{band} x N_{SCA} x N_{det}	Long collect statistic Characterization Stat Char Q_{min}	Float
Interval Segment Signal StDev	$Sigma\_Q_{(B,S,D}$	DN or W/m^2 sr µm	N_{band} x N_{SCA} x N_{det}	Long collect statistic Characterization Stat Char σ	Float
Number of frames	NBR_of_frames			Long collect statistic Characterization Stat Char *number of frames*	Long
Interval segment ID or segment start line (line time tag)	Seg_ID	s	N_{Seg}	Long collect statistic Characterization Acquisition time for the first frame of each segment t_1	Float
Scene Type (i.e O* or D*)	Stype		N_{band}	DB L0Rp Image file	Char
Processing Step (i.e., before or after Gain application)	Pstep		N_{band}	Histogram Stat Char Position in processing flow (RPS Level)	String
Segment window size (lines)	Seg_win			Analyst input /pre-processing default set for 60sec duration – i.e., 4200 lines	Int
Relative gains	r		N_{band} x N_{SCA} x N_{det}	CPF	Float
Impulse Noise Pixels Locations	LM1			LM	Int
Saturated Pixel Mask	LM2			LM	Int
Dropped Frames Mask	LM3			LM	Float
Inoperable Detector List	Dinop			CPF	Float

4.6.6.3 Outputs

Outputs will mimic the provided segmentation of the input data (i.e., even if actual data was given as a single continuous collect since its inputs from the special long hist_stat processing will be chopped into multiple segments then the output DB will include

results per segment – only in this case the segments are artificially produced by a moving window with in one or more mission data files).
Per interval segment populate these outputs.

The algorithm will need to process the data from all segments before the analyst could assess the stability results for the full interval.

For Segment I the output to DB is:

Description	Symbol	Units	Level	Target	Type
Signal Variability, SCA average	$\overline{\Delta Q_{B,S}}$	DN or W/m2 sr µm	N_{band} x N_{SCA}	DB (Bias, Gains & OBC L1R Stability)	Float
L1r Product Variability, SCA average	$\overline{\Delta_{B,S}}$	%	N_{band} x N_{SCA}	DB (Gains & OBC L1R Stability)	Float
SNR Variability, per-detector	$\overline{\Delta_{B,S,D}}$	%	N_{band} x N_{SCA} x N_{det}	DB (Gains & OBC L1R Stability)	Float
Scene Type (i.e O* or D*)	Stype		N_{band}	DB	Char
Processing Step	Pstep		N_{band}	DB	String
Interval segment ID or segment start line (line time tag)	Seg_ID		N_{Seg}	DB	Float
Segment window size (lines)	Seg_win			DB	Int

4.6.6.4 Options

- Trending On/Off Switch: If trending is Off, output parameters are written to a text file.
- Comp stat switch: ON – compute statistics, Off – import stat data from hist_stat (default)
- Collect type ID – Cont 36min / 1.5 orbit data
- Window sizes – full (default), 4200, User selected between 100-64000 ; if collect type ID is Cont 36min the window size parameter will be used to define the segment size ; If Comp-stat switch is Off segment window size will be imported from hist_stat.
- Processed Blind_band – Yes / No

4.6.6.5 Procedure

Note that in order to characterize TIRS radiometric stability, we call this algorithm at least 3 times: Once to process long Deepspace data, and twice to process the data for the closest long OBC collects (once using the net linearized response L0rc1 intermediate Cal product and once using the radiometric corrected L1R product.)

Pre-Processing (creating sub segments generating the input data needed and computing Hist_stat info for each segment) The preprocessing brings either Collect type into a fixed input format.

1- Pre-Process the data (if in multi-mission data files continuous or multi-segments of Cal-sequence)

Check if Collect type option is Cont 36min data

If No – get sequence of 30 scenes IDs to processes through this algorithm description

If yes chop the 36 min data into smaller segments in the size of Window_size input parameter

i.e., $Q_{interval}(b,s,d,l)$ - > { $Q_1(b,s,d,l)$; $Q_2(b,s,d,l)$; …. $Q_n(b,s,d,l)$ }

where the last line in Q_n is the last line in $Q_{interval}$ and all segment are set to equal size (Seg_win).

For a default window size there should be 36 scene IDs generated.

For all segments produce or obtain from DB the input histogram information needed for the algorithm.

2- On each scene or segment (30 or 36 segments) within the interval proceed to the following steps of 3-8

Calculating statistics (possibly can be retrieved from Hist Stat DB for the scene or segment)

3- Based on the Labeled Mask and detector operability list omit those detectors when calculating SCA level averages.

4- Populate histogram statistics metric per detector in the relevant variables and calculate the average across all detectors within an SCA.

a. $\overline{Q}_{(B,S,D)} = Histogram\_per\_\det ector\_mean\_signal\ \overline{Q}$ (1)

b. $Max\_\overline{Q}_{(B,S,D)} = Histogram\_per\_\det ector\_Max\_signal\ Q_{max}$ (2)

c. $Min\_\overline{Q}_{(B,S,D)} = Histogram\_per\_\det ector\_Min\_signal\ Q_{min}$ (3)

d. $Sigma\_\overline{Q}_{(B,S,D)} = Histogram\_per\_\det ector\_StDev\ \sigma$ (4)

e. $\overline{Q}_{B,S} = mean(\overline{Q}_{B,S,D})$ (5)

f. Get *number_of_frames* for the interval segment

g. Get Interval segment ID from input (this will be related to the segment's first line time tag)

h. Note that calculations should only include operable and in-spec detectors. (i.e., ignore pixels flagged as inoperable, saturated, dropped frame, impulse or fill.)

Calculating Stability of Signal

5- For each detector, calculate the 1-sigma the along-track variation in the image that is at the length of the segment window or segment duration.

$$\Delta Q_{(B,S,D)} = 1 \times Sigma\_\bar{Q}_{(B,S,D)} \quad (6)$$

Processing data to generate outputs

6- Calculate the SCA average variability, i.e, the average across all detectors within each SCA:

$$\overline{\Delta Q_{B,S}} = mean(\overline{\Delta Q_{B,S,D}}) \quad (7)$$

7- For TIRS deep space data, record the SCA mean variabilities ($\overline{\Delta Q_{B,S}}$) of every segment or scene within the interval to the database or output file (along with other specified outputs in the output table). This is the end of the algorithm for processing TIRS deep space data.

8- For TIRS OBC data process and record what is done up to step 7 and also do this :

a. Calculate the per-detector variability in terms of percent change. It illustrates the impact of individual detectors on the overall radiometric stability.

ii. $$\overline{\Delta_{B,S,D}} = r_D * \frac{\overline{\Delta Q_{B,S,D}}}{\overline{Q}_{B,S,D}} * 100\% \quad (8)$$

iii. where r_D is the relative linear gain used for conversion to at aperture radiance from CPF for L0Rc data and 1.0 for L1R data.

b. Calculate the SCA average variability in terms of percent gain change:

ii. $$\overline{\Delta_{B,S}} = \frac{\overline{\Delta Q_{B,S}}}{\overline{Q}_{B,S}} * 100\% \quad (9)$$

iii. $\overline{Q}_{B,S}$ should be calculated using the same list of detectors used to produce the mean signal in equation (5)

c. For L0Rc data, write the per-detector percent variability ($\overline{\Delta_{B,S,D}}$) and the SCA average variability ($\overline{\Delta_{B,S}}$) of every segment or scene within the interval to the database or output file (along with other specified outputs in the output table).

d. For L1R data, write the SCA average percent variability ($\overline{\Delta_{B,S}}$) of every segment or scene within the interval to the database or output file (along with other specified outputs in the output table).

e. For L1R if output to file selected also write to file the per-detector variability ($\overline{\Delta_{B,S,D}}$) and the SCA Radiance mean variability ($\overline{\Delta Q_{B,S}}$)of every segment or scene within the interval (along with other specified outputs in the output table).

9- Once all segments or scenes had been processed analyst will work offline to extract information from DB to generate various plots and test correlations to telemetry data. Most of this follow-on analysis is not automated or strictly defined by this algorithm but it may ultimately include plots of the SNR variability over the interval and use such plots to identify detectors are 5 time less stable than the overall band stability characteristics assisting with detector operability characterization. Other likely plots are plots similar to lamp response trending – trending mean response Vs. time and stdev of response Vs. time in each segment over the 36 minute or longer interval.

4.6.7 TIRS Second Linearization

4.6.7.1 Background/Introduction

Conversion to radiance (L1R) occurs in several steps. First the linearized dark and background responses are subtracted from the linearized L0r data. This first linearization, which is derived from integration time sweep data, is designed to account for non linearity in response of the electronics of the system. Then, an additive linearity correction is applied to the data. This second linearization, which is derived from black body calibrator sweeps, is designed to account for non-linearity in the response of the individual detectors. Finally, a gain and gain offset coefficient are applied resulting in the L1R product in units of radiance.

This algorithm applies the look-up table (LUT) based additive linearity correction to the data.

4.6.7.2 Inputs

Description	Level	Source	Type
Linearized, Background Subtracted Scene ($Q_{lin1,BS}$)	N_{band} x N_{SCA} x N_{det} x N_{frame}	TIRS Background Subtraction	Float
Second Linearization Look Up Table (LUT)	N_{band} x N_{SCA} x N_{det} x N_{Level}	File	Float

4.6.7.3 Outputs

Description	Level	Target	Type
Linearized Scene (Q_{lin2})	N_{band} x N_{SCA} x N_{det} x N_{frame}	Gain Application	Float

4.6.7.4 Procedure

For each pixel in the scene:

1. Look up the two nearest additive linearization correction values *r* for a the two nearest values of the given pixel value in $Q_{lin1,BS}$.

$$[r_1(f,d,s,b), r_2(f,d,s,b)] = LUT\left(\left[Q_{2,lin1,BS}(f,d,s,b), Q_{1,lin1,BS}(f,d,s,b)\right]\right) \quad (1)$$

   where *f* is frame, *d* is detector, *s* is SCA and *b* is band.

2. Determine the value of *r* using linear interpolation.

$$r(f,d,s,b) = r_1(f,d,s,b) + \left(r_2(f,d,s,b) - r_1(f,d,s,b)\right)\left(\frac{Q_{lin1,BS}(f,d,s,b) - Q_{1,lin1,BS}(f,d,s,b)}{Q_{2,lin1,BS}(f,d,s,b) - Q_{1,lin1,BS}(f,d,s,b)}\right) \quad (2)$$

   Note: If $Q_{lin1,BS}$ is outside of the range of the upper or lower values in the LUT, then *r*=0 for that pixel.

3. Add *r* to $Q_{lin1,BS}$

$$Q_{lin2}(f,d,s,b) = Q_{lin1,BS}(f,d,s,b) + r(f,d,s,b) \quad (3)$$

4.6.8 TIRS Radiance to Brightness (Apparent) Temperature

4.6.8.1 Background/Introduction

Image data that has been converted to spectral radiance may be converted into apparent temperature (i.e. - the temperature of a blackbody with emissivity of 1.0 that would produce the spectral radiance when integrated over the band relative spectral response).

For a given calculated spectral radiance for a particular band, a look-up table (LUT) is created that relates the temperature of a blackbody (T in Deg K) to the integrated spectral radiance in the band (L in W/m^2-sr-mu). The spectral radiance is calculated through the Planck blackbody equation and then multiplying by the spectral response function of the band per wavelength and integrating over wavelength as,

$$L_b(T) = \frac{\int B(\lambda,T)\cdot R_b'(\lambda)\cdot d\lambda}{\int R_b'(\lambda)\cdot d\lambda} \quad (1)$$

Where $L_b(T)$ is the integrated spectral radiance for the spectral band at a given blackbody temperature, $R'_b(\lambda)$ is the relative spectral response of the band, and $B(\lambda,T)$ is the Planck radiance at the given blackbody temperature T.

The Look-Up Table (LUT), example in Figure 4-105, stores values of spectral radiance and corresponding apparent blackbody temperature between 240K and 360K allowing for conversion between the two, for each TIRS band, in steps of 0.01K.

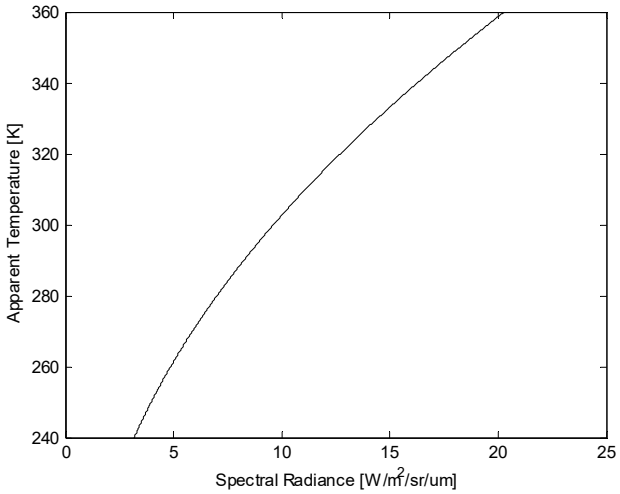

Figure 4-105. Example TIRS-1 10.8 µm Band Look-Up Table (LUT)

4.6.8.2 Inputs

Description	Symbol	Units	Level	Source	Type
Planck Function	$B(\lambda,T)$	W/m^2 sr µm	N_{band}	Derived	Float
Relative Spectral Response	$R'_b(\lambda)$		N_{band}	CPF	Float

4.6.8.3 Outputs

Descriptions	Symbol	Units	Level	Source	Type
Integrated spectral radiance	$L_b(T)$	W/m^2 sr µm	N_{band}	Derived	Float
Look Up Table (Rad vs Temp)	LUT	W/m^2 sr µm vs Deg K	N_{band}	Anc File	Float

4.6.8.4 Procedure

1. Read $R'_b(\lambda)$

For each TIRS-2 band

2. Generate LUT between 180 K to 360 K to cover possible scene temperatures:

   For T from 180 to 360, 0.1 increments
   a. Calculate $B(\lambda,T)$
   b. Calculate $L_b(T)$
   c. Print T, $L_b(T)$ *to LUT*

3. convert recorded band radiance (L_bandmeasured) into apparent temperature (T_apparent) via LUT:

   T_apparent = LUT(L_bandmeasured)

4.6.9 Landsat 8 TIRS Stray Light Correction

4.6.9.1 Background

Image data from Landsat 8 TIRS bands 10 and 11 are contaminated by stray light artifacts caused by radiance sources entering the instrument's optical system from out of its ±7.5 degrees field of view and adding an unwanted non-uniform signal onto the focal plane detectors. These scene-dependent effects manifest themselves as a banding artifact in TIRS imagery and as a varying absolute radiometric error. The banding appears as low-frequency scene-dependent variation in at-sensor radiance across the field-of-view. The TIRS-derived at-sensor radiance is always up to about 8% larger than it should be, and that error varies seasonally. Specially designed scans of the moon along with a detailed optical model of the TIRS optical system yielded maps describing the locations (angles) and magnitudes (weights) of the stray light sources for each single detector [1]. These stray light source maps allow the total amount of the out-of-field stray light radiance signal for every detector to be computed and then subtracted from the Level 1 radiance product resulting in large reduction of the stray light artifacts in TIRS imagery [2].

The per-detector stray light maps present the key aspect of this algorithm. The out-of-field locations of stray light sources are represented by a set of line-of-sight vectors for each detector (each detector may have tens of vectors). Figure 4-106 illustrates the stray light map vectors for one example detector. Along with attitude telemetry data coming from the observatory, these vectors are used to determine geographical locations (latitude/longitude) of the stray light sources for every detector and every image frame for a given TIRS image [2,3]. To estimate the actual radiances at these stray light source locations, out-of-field thermal data coincidentally acquired by a wide-field-of-view sensor with similar design characteristics would be the ideal source [2].

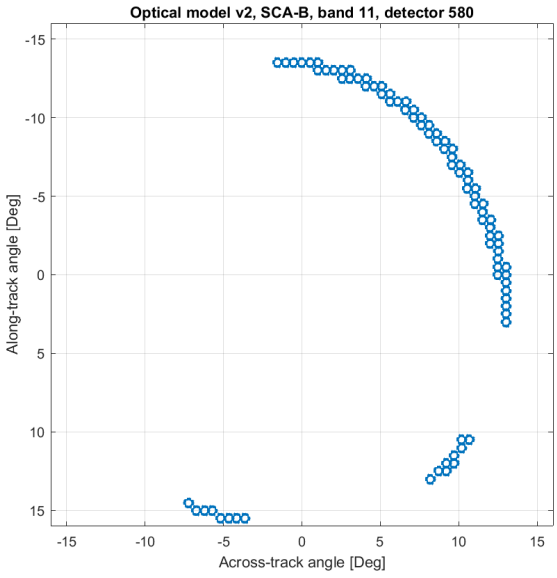

Figure 4-106. Stray light map locations expressed as angles from the optical axis for one example detector [3]

Since such data source does not really exist and to avoid processing and calibration limitations associated with using image data from a potentially suitable external wide-field-of-view sensor, such as GOES, this algorithm uses the TIRS path interval itself as a substitution to obtain the out-of-field radiances for each stray light vector. The sampling process uses the latitude/longitude locations from the stray light map data to determine the locations in the TIRS path interval from which to retrieve the radiance values. Note that for vector locations that fall outside the path, radiance values from the closest pixels at the edge of the interval are used [2,3]. This sampling option is known as the "TIRS-on-TIRS" option and is illustrated for an example detector in Figure 4-107.

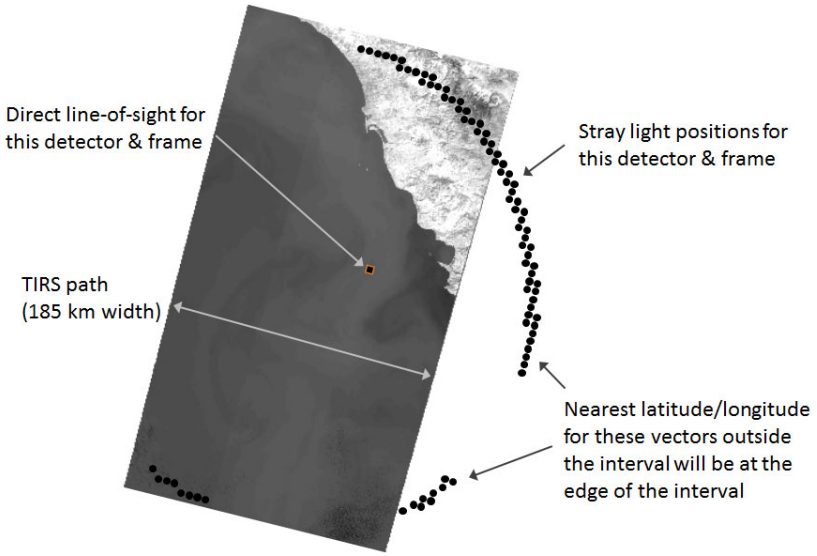

Figure 4-107. Stray light map locations projected onto the ground as a series of latitude/longitude points for one example detector. The radiance values at the nearest locations to these points are used in the algorithm [3].

Once radiances are obtained for each stray light vector for each detector and each image frame of the Earth scene, they are multiplied by the weighting values associated with each vector in the stray light map. The weighting values describe the relative magnitude of the stray light signal contribution of each location (vector). The weighted radiances for each detector are then summed to yield the total stray light signal for the detector. Since the weights are relative values for each detector individually, the summed values are scaled to true stray light signal radiances through the provided linear coefficients for each detector [2,3]. After performing this calculation (computing latitude/longitude locations, sampling the radiances at these locations, multiplying by the associated weights, summing, and scaling by the linear coefficients) for each detector and each image frame (i.e. – image line), the resulting 'image' consists of the estimated extra stray light signal for the TIRS scene. This stray light image is then subtracted from the L1 radiances to remove the stray light artifacts [2,3].

4.6.9.2 Input

Descriptions	Symbol	Units	Level	Source	Type
Scene (L1R): TIRS Earth data after saturated pixel replacement and inoperable detector fill	L	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{bands} x N_{SCAs} x $N_{detectors}$ x N_{frames}		Float
TIRS three-scene sub-interval (consisting of the scene of interest and one scene acquired before and one after it)		DN	3 x N_{bands} x N_{SCAs} x $N_{detectors}$ x N_{frames}		Unsigned Integer
Stray light map vectors & weights for each detector	$w_{i,j}$	unitless	N_{bands} x N_{SCAs} x $N_{detectors}$ x $N_{vectors}$	Vector and weight file*	Float
Linear scaling coefficient - slope	a_j	unitless	N_{bands} x N_{SCAs} x $N_{detectors}$	Linear coefficient file**	Float
Linear scaling coefficient - intercept	b_j	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{bands} x N_{SCAs} x $N_{detectors}$	Linear coefficient file**	Float

*** vector and weight file:** tirs_stray_light_vectors.csv
**** linear coefficient file:** TIRS_radiance_coefficients.csv

4.6.9.3 Output

Descriptions	Symbol	Units	Level	Target	Type
Stray light image	S	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{bands} x N_{SCAs} x $N_{detectors}$ x N_{frames}		Float
Scene (L1R-corrected)	L_c	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{bands} x N_{SCAs} x $N_{detectors}$ x N_{frames}		Float

4.6.9.4 Procedure

For each Landsat Earth scene and both TIRS spectral bands, *b*:

4. Get a three-scene sub-interval centered on the scene that is being processed:
    a. Subsetter provides three-scene sub-interval as L0Rp
    b. Run the ancillary data preprocessing and Line-Of-Sight (LOS) model creation processing steps on the three-scene sub-interval.
    c. Process the sub-interval to L1R.
    d. Create a grid for the TIRS band.
    e. Resample the L1R to an L1G systematic image using nearest neighbor resampling. Leave L1G in units of radiance (not DN).

5. Create stray light image:
    a. Read the TIRS stray light vectors from the Vector and weight file.
    b. Read linear scaling coefficients from the Linear coefficient file.

c. Read LOS model.
d. Read TIRS sub-interval L1G data created in step 1 and blur the 100 m data down to 6 km resolution (to more closely match optical model resolution).
e. Read the original L0R metadata to get the extents of the target scene in L0R/L1R space.
f. For each SCA, *s*, detector, *j*, and frame, *f*, of the scene to be corrected, in L0R/L1R space:
    i. For each stray light vector *i* associated with the detector *j*:
        1. Convert stray light vector to lat/long location (stripped down version of ias_los_model_input_line_samp_to_geodetic).
        2. Convert the lat/long to the L1G map projection.
        3. Extract radiance from the lat/long location, L_i. If map coordinates map to a non-fill pixel in the L1G image (from step 2d), use radiance from that pixel. Otherwise, find closest available pixel in the L1G image by finding the shortest distance from the map coordinate, and use radiance from that pixel.
    ii. Weight and sum the found radiances, L_i, associated with the stray light locations.

$$L_S(b,s,j,f) = \sum_i w_{i,j} L_i$$

    iii. Apply the linear scaling coefficients to the summed radiance for the detector.

$$S(b,s,j,f) = a_j L_S(b,s,j,f) + b_j$$

g. Result is a stray light image, *S(b)* for the TIRS scene of interest in L1R space (in units of radiance).

6. Correct TIRS data:
    a. Create new L1R that only has the single scene of TIRS data in it
    b. Read L1R stray light image from step 2g.
    c. Subtract stray light image from the TIRS L1R data.

$$Lc(b) = L(b) - S(b)$$

    d. Return to nominal processing flow (i.e.- L1R is resampled to L1T through standard process).

4.6.9.5 References

[1] Montanaro, M., Gerace, A., Lunsford, A., & Reuter, D. (2014). Stray light artifacts in imagery from the Landsat 8 Thermal Infrared Sensor. *Remote Sensing*, 6(11), 10435-10456. doi:10.3390/rs61110435

[2] Montanaro, M., Gerace, A., & Rohrbach, S. (2015). Toward an operational stray light correction for the Landsat 8 Thermal Infrared Sensor. *Applied Optics*, 54(13), 3963-3978. doi: 10.1364/AO.54.003963
[3] Gerace, A., & Montanaro, M. (2017). Derivation and validation of the stray light correction algorithm for the Thermal Infrared Sensor onboard Landsat 8. *Remote Sensing of Environment*, 191, 246-257. doi: 10.1016/j.rse.2017.01.029

4.6.10 TIRS Radiance Bias Removal

4.6.10.1 Background

Stray light correction applied to TIRS data may introduce systematic biases into the L1 radiance products. This algorithm is applied to stray light corrected products to remove those potential biases in radiance units from two TIRS spectral bands (the blind band excluded).

Per band, per SCA and per detector bias correction values are determined from analysis of the stray light corrected products and provided in the CPF. Bias removal is accomplished by subtracting these values from each pixel of the input image. If the bias correction needs to increase the radiance, the bias should be defined as a negative value in the CPF.

This algorithm is not needed for Landsat 9, but the L9 CPF should still include the groups and parameters for consistency with the L8 CPF and in case it is determined to be needed. All the parameter values will be set to zero in the CPF and the algorithm will not be included in the L9 processing flows.

4.6.10.2 Input

Descriptions	Symbol	Units	Level	Source	Type
Scene (L1R) TIRS stray light corrected Earth data	L	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{bands} x N_{SCAs} x $N_{detectors}$ x N_{frames}		Float
TIRS Band Radiance Bias	B_b	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{bands}	CPF	Float
TIRS SCA Radiance Bias	B_s	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{bands} x N_{SCAs}	CPF	Float
TIRS Detector Radiance Bias	B_d	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{bands} x N_{SCAs} x $N_{detectors}$	CPF	Float

4.6.10.3 Output

Descriptions	Symbol	Units	Level	Target	Type
Scene (L1R-corrected)	L_c	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{bands} x N_{SCAs} x $N_{detectors}$ x N_{frames}		Float

4.6.10.4 Options

Work order parameters should allow the operator to separately choose whether or not to apply per band, per SCA, and/or per detector radiance bias correction.

4.6.10.5 Procedure

1. For each detector, d, for all TIRS imaging bands, b, (the blind band excluded) and SCAs, s, calculate the total radiance bias as:

$$bias(b,s,d) = B_b(b) + B_s(b,s) + B_d(b,s,d)$$

2. Remove the bias from each frame, f, of the stray light corrected scene:

$$L_c(b,s,d,f) = L(b,s,d,f) - bias(b,s,d)$$

4.6.11 TIRS Radiance Gain Application

4.6.11.1 Background

Stray light correction applied to TIRS data may introduce additional banding or striping in the L1 radiance products or minor change in band average gains. This algorithm is applied to stray light corrected products to adjust for changes in detector gain levels of two TIRS spectral bands (the blind band excluded) in Earth scenes.

Per band, per SCA and per detector gain correction values are determined from analysis of the stray light corrected products and provided in the CPF. Radiance gain correction is accomplished through dividing of each pixel of the input image by those CPF values. If any of the gain correction levels is not needed (i.e. if there is no need for per detector radiance gain correction), the parameter values in the CPF are set to 1.0 for that correction.

This algorithm is not expected to be needed for Landsat 9, but the L9 CPF should still include the groups and parameters for consistency with the L8 CPF and in case it is determined to be needed. All the parameter values will be set to 1.0 in the CPF and the algorithm will not be included in the L9 processing flows.

4.6.11.2 Input

Descriptions	Symbol	Units	Level	Source	Type
Scene (L1R) TIRS stray light corrected Earth data	L	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{bands} x N_{SCAs} x $N_{detectors}$ x N_{frames}		Float
TIRS Band Radiance Gain	G_b	Unitless	N_{bands}	CPF	Float
TIRS SCA Radiance Gain	G_s	Unitless	N_{bands} x N_{SCAs}	CPF	Float
TIRS Detector Radiance Gain	G_d	Unitless	N_{bands} x N_{SCAs} x $N_{detectors}$	CPF	Float

4.6.11.3 Output

Descriptions	Symbol	Units	Level	Target	Type
Scene (L1R-corrected)	L_c	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{bands} x N_{SCAs} x $N_{detectors}$ x N_{frames}		Float

4.6.11.4 Procedure

1. For each detector, d, for all TIRS imaging bands, b, (the blind band excluded) and SCAs, s, calculate the total radiance gain as:

$$G_{total}(b,s,d) = G_b(b) \times G_s(b,s) \times G_d(b,s,d)$$

2. Apply the total gain to each frame, f, of the stray light corrected scene:

$$L_c(b,s,d,f) = \frac{L(b,s,d,f)}{G_{total}(b,s,d)}$$

4.6.12 TIRS Radiance Histogram Statistics Characterization

4.6.12.1 Background/Introduction

This algorithm is an occurrence of the general purpose “Histogram Statistics Characterization” algorithm in radiance space of the radiometric processing flow. For Landsat-8, this algorithm is run after the stray light correction and the TIRS radiance bias removal algorithms in the radiometric processing flow. For Landsat-9, in absence of stray light correction, this algorithm will be run after the gain application algorithm. This algorithm supports characterization of all active TIRS detectors, including the inoperable ones, but excluding the blind band, by computing statistics from Earth scenes: minimum, maximum, mean and standard deviation The Earth scene per SCA and per band means and standard deviations are also computed. All results are stored in the database and used primarily to characterize residual striping and estimate TIRS residual relative gains.

4.6.12.2 Inputs

Description	Symbol	Units	Level	Source	Type
Scene (L1R) TIRS Earth data	L	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA} x N_{det} x N_{frame}		Float
Impulse Noise Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
Dropped Frame Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int
Saturated Pixel Locations			N_{band} x N_{SCA} x N_{det} x N_{frame}	LM	Int

4.6.12.3 Outputs

Description	Symbol	Units	Level	Target	Type
Detector Minimum response	Q_{min}	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA} x N_{det}	Db	Float
Detector Maximum response	Q_{max}	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA} x N_{det}	Db	Float
Detector Mean	$\overline{Q}$	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA} x N_{det}	Db	Float
Detector Standard deviation	σ	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA} x N_{det}	Db	Float
Number of valid frames for a detector	N_{valid_pixels}	Pixels	N_{band} x N_{SCA} x N_{det}	Db	Int
SCA Mean	$\overline{Q}_{SCA}$	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA}	Db	Float
SCA Standard deviation	σ_{SCA}	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band} x N_{SCA}	Db	Float
SCA average number of frames	N_{SCA_frames}		N_{band} x N_{SCA}	Db	Float
Band Mean	$\overline{Q}_{band}$	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band}	Db	Float
Band Standard deviation	σ_{band}	$\frac{W}{m^2 \cdot sr \cdot \mu m}$	N_{band}	Db	Float
Band average number of frames (for Earth scenes only)	N_{band_frames}		N_{band}	Db	Float
Position in processing flow (RPS level)			1	Db	String
Linearization LUT version			1	Db	String

Note: This instance of the algorithm will only be run for TIRS bands

4.6.12.4 Options

Typically, these data will be stored in the characterization database. For stand-alone processing, the individual detector statistics may be output to a summary report with a header containing start date and time of acquisition, end date and time of acquisition, processing date and time, calculated frame rate, filename and entity ID. A report generation should be selectable in work order.

4.6.12.5 Procedure

For each active detector, *d*, including inoperable detectors , for all TIRS bands and SCAs, except the blind band:

Obtain number of valid pixels, $N_{valid\_pixels}\ (d)$, in image or collect. Pixels identified in the label mask as saturated, impulse noise affected, or parts of dropped frames, as well as filled pixels used to generate detector offsets and pixels corresponding to the Number of frames to be skipped at the top and bottom of the image, are considered invalid and need to be taken out of calculations. The symbol *l* is used to denote the list of valid pixels.

1. Find the minimum of valid pixel values, *l*:

$$Q_{\min}(d) = \min\left(Q\left(l,d\right)\right)$$

2. Find the maximum of the valid pixel values, *l*:

$$Q_{\max}(d) = \max\left(Q\left(l,d\right)\right)$$

3. Calculate mean as:

$$\overline{Q}(d) = \frac{1}{N_{valid\_pixels}(d)} \sum_{l=1}^{N_{valid\_pixels}} Q(l,d)$$

4. Calculate standard deviation as:

$$\sigma(d) = \sqrt{\frac{1}{N_{valid\_pixels}(d)} \sum_{l=1}^{N_{valid\_pixels}} (Q(l,d) - \overline{Q}(d))^2}$$

5. Save results to the database. Generate a report file, if that option is selected in work order.

For each image band (TIRS bands 10 and 11):

6. Calculate the SCA mean as:

$$\overline{Q}_{SCA} = \frac{\sum_{d=1}^{N_{detectors}} N_{valid\_pixels}(d) \times \overline{Q}(d)}{\sum_{d=1}^{N_{detectors}} N_{valid\_pixels}(d)}$$

7. Calculate the SCA standard deviation as:

$$\sigma_{SCA} = \sqrt{\frac{\sum_{d=1}^{N_{detectors}} N_{valid\_pixels}(d)\left(\sigma(d)^2 + \overline{Q}(d)^2\right) - \left(\sum_{d=1}^{N_{detectors}} N_{valid\_pixels}(d)\right) \times \overline{Q}_{SCA}{}^2}{\sum_{d=1}^{N_{detectors}} N_{valid\_pixels}(d)}}$$

8. Calculate the average number of frames for SCA as:

$$N_{SCA\_frames} = \frac{\sum_{d=1}^{N_{detectors}} N_{valid\_pixels}(d)}{N_{detectors}}$$

9. Calculate the band mean as:

$$\overline{Q}_{band} = \frac{\sum_{s=1}^{N_{SCAs}} N_{SCA\_frames}(s) \times \overline{Q}_{SCA}(s)}{\sum_{s=1}^{N_{SCAs}} N_{SCA\_frames}(s)}$$

10. Calculate the band standard deviation as:

$$\sigma_{band} = \sqrt{\frac{\sum_{s=1}^{N_{SCAs}} N_{SCA\_frames}(s)\left(\sigma_{SCA}(s)^2 + \overline{Q}_{SCA}(s)^2\right) - \left(\sum_{s=1}^{N_{SCAs}} N_{SCA\_frames}(s)\right) \times \overline{Q}_{band}{}^2}{\sum_{s=1}^{N_{SCAs}} N_{SCA\_frames}(s)}}$$

11. Calculate the average number of frames for band as:

$$N_{band\_frames} = \frac{\sum_{s=1}^{N_{SCAs}} N_{SCA\_frames}(s)}{N_{SCAs}}$$

12. Save these scene summary statistics to the database

4.7 Level 2 Algorithms

Level 2 Landsat Science Products are generated from algorithms that calculate Top of Atmosphere Reflectance (TOA-R), Top of Atmosphere Brightness Temperature (TOA-BT), Surface Reflectance (SR), and Surface Temperature (ST). All Level 2 algorithms are subject to two primary requirements that constrain which Level 1 scenes are eligible for production, as listed below.

- Solar zenith angle is 76 degrees or less
- Input auxiliary data files are available

4.7.1 Level 2 Auxiliary Preprocessing Algorithm

4.7.1.1 Background/Introduction

The Land Surface Reflectance Code (LaSRC) requires auxiliary inputs for ozone and climate variables. The Single Channel Landsat Surface Temperature (ST) algorithm is also based on several auxiliary inputs. The auxiliary inputs for each algorithm are described below.

4.7.1.2 Auxiliary Inputs

4.7.1.2.1 LaSRC Auxiliary Input Source

Description	Source
MODIS Aqua/Terra CMA	Pulled from the https site
MODIS Aqua/Terra CMG	Pulled from the https site

The Moderate Resolution Imaging Spectroradiometer (MODIS) Terra/Aqua Aerosol Optical Thickness (MO/YD09CMA) and Surface Reflectance (MO/YD09CMG) products provide daily measurements of air temperature, water vapor, aerosol optical thickness, and ozone in Climate Modeling Grid (CMG) HDF files. The MODIS data are pulled from https://ladsweb.modaps.eosdis.nasa.gov and the following directories are used for each instrument and data type:

```
TERRA_CMA = '/archive/allData/6/MOD09CMA/'
TERRA_CMG = '/archive/allData/6/MOD09CMG/'
AQUA_CMA = '/archive/allData/6/MYD09CMA/'
AQUA_CMG = '/archive/allData/6/MYD09CMG/'
```

MODIS products are of the filename M[O|Y]D09CM[A|G].A{year}{doy}.006.*.hdf.

MODIS Products	Grid (lat/lon)
Daily aerosol optical thickness	0.05° x 0.05° 90° N to 90° S and 90° W to 90° E
Daily global surface reflectance	0.05° x 0.05° 90° N to 90° S and 90° W to 90° E

The following parameters are extracted from MODIS products as input to the LaSRC algorithm.

Product	File Type	Type
Water Vapor	HDF	16-bit integer
Ozone SDSs	HDF	8-bit integer

4.7.1.2.2 Single Channel Surface Temperature Auxiliary Input Source

Description	Source
MERRA-2	Pulled from the http site
FP-IT	Pulled from the http site
ASTER GED	Pulled from the LP DAAC

The Modern-Era Retrospective analysis for Research and Applications, Version 2 (MERRA-2) and Goddard Earth Observing System Model, Version 5 (GEOS-5) Forward Processing – Instrument Team (FP-IT) products are atmospheric reanalysis netCDF files which contain numerous values, including several used by the Surface Temperature algorithm, including air temperature (T), geopotential height (H), and specific humidity (QV). The Surface Temperature procedure only uses one of the 2 datasets for a given run.

The MERRA-2 data are pulled from https://goldsmr5.gesdisc.eosdis.nasa.gov/data/MERRA2/M2I3NPASM.5.12.4/{year}/{month}/MERRA2_{1..4}00.inst3_3d_asm_Np.{year}{month}{day}.nc4, where {1..4} is the production stream used for a given date, and "00" is the version number.

The FP-IT data are pulled from https://goldsfs1.gesdisc.eosdis.nasa.gov/data/GEOS5/DFPITI3NPASM.5.12.4/{year}/{dayofyear}/GEOS.fpit.asm.inst3_3d_asm_Np.GEOS5124.{year}{month}{day}_{hour}00.V01.nc4.

The Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER) Global Emissivity Dataset (GED) is a static dataset. The Surface Temperature procedure uses ASTER GED emissivity mean bands 13 and 14, emissivity standard deviation bands 13 and 14, as well as calculated Normalized Difference Vegetation Index (NDVI) mean band. The ASTER GED data are pulled from the Land Processes Distributed Active Archive Center (LP DAAC) using dataset /ASTT/AG100.003/2000.01.01.

Product	Parameter	File Type	Grid (lat/lon)	Daily Values (Z)	Data Type
MERRA-2	Temperature, specific humidity, geopotential height	NetCDF	0.625º x 0.5º 90° S to 90° N and 179.375º E to -180.0º W	0, 3, 6, 9, 12, 15, 18, 21	Float
FP-IT	Temperature, specific humidity, geopotential height	NetCDF	0.625º x 0.5º 90° S to 90° N and 179.375º E to -180.0º W	0, 3, 6, 9, 12, 15, 18, 21	Float
ASTER GED	Emissivity mean, emissivity standard deviation, NDVI mean	HDF5	1.0º x 1.0º main land masses	Static dataset	Int16

4.7.1.3 Procedure

4.7.1.4 LaSRC Auxiliary Preprocessing

4.7.1.4.1 Processing MODIS Datasets

The updatelads.py script handles the processing of the MODIS datasets. It supports daily and quarterly processing of data products.

1. Daily processing – processing products for the current year
    a. If (current month <= 30 days of start of year), then process data from the previous year as well
2. On-Demand processing – processing products within a specified date range

MODIS products for the desired date range are temporarily downloaded to a temporary directory for processing. The products are fused and repackaged into daily products with only water vapor and ozone values. The original products are removed from the temporary directory. The output daily HDF files are written to $L8_AUX_DIR/LADS/{year}.

1. combine_l8_aux_data executable
    a. Reads daily Terra and Aqua CMA and CMG products
    b. Fuses products into a single product
        i. Terra product is the baseline for both the coarse resolution ozone (from CMG) and water vapor (from CMA) values
        ii. Aqua product is used to fill in the missing / NoData values in the Terra products
    c. If either the Terra or Aqua products are missing, then the other product is used for the output values
    d. Any remaining fill values are interpolated to generate a full output HDF product
    e. Both the ozone and water vapor variables are written as Science Datasets (SDS) to the output HDF file.

4.7.1.4.2 Time Lag in Data Availability

The MODIS products have a 2-3 day time lag between acquisition and availability via http download. To balance timeliness with the desire to wait for both Terra and Aqua products to be available, run the updatelads.py script as a nightly script using the --today switch to designate daily processing. This script will download the available MODIS products from their specific http site and package those into the output HDF files which are then used by LaSRC for processing to surface reflectance.

4.7.1.5 Single Channel Landsat Surface Temperature Auxiliary Preprocessing

4.7.1.5.1 Processing MERRA-2 and FP-IT Datasets

Single Channel Landsat Surface Temperature (ST) processing requires atmospheric data near the time and location of the Landsat scene. The implemented algorithm uses the MERRA-2 or FP-IT dataset to satisfy this requirement. The ST software is supported by two management scripts used to download MERRA-2 and FP-IT data required by the procedure:

- st_aux_merra_from_NASA_archive.py – extracts the MERRA-2 variables ST requires (height, temperature, specific humidity), and stores them in a local archive. It can use a specific date or a date range.
- st_aux_geos_from_NASA_archive.py – extracts the FP-IT variables ST requires (height, temperature, specific humidity), and stores them in a local archive. It can use a specific date or a date range.

The tools allow selection of dates or date ranges to retrieve or use of default values for the dates.

The auxiliary data tools are controlled by the level2_auxiliary.conf file which defines values such as the location of the remote archive, the format of the files in the remote archive, the location of the local archive directory, the archive directory and filename formats, the default date range to retrieve, and others.

The production ST software assumes some of the default configuration settings such as archive directory structure are used. With default configuration file settings, after the tools are run, the MERRA-2 or FP-IT data is packaged by these tools to be in a specific directory format. For MERRA-2 the highest level in the directory format is the year, followed by the month. For FP-IT, the highest level in the directory format is the year, followed by the month, followed by the day of month. For example, the MERRA-2 directory for June 2016 is "$ST_MERRA_AUX_DIR/2016/06", and the FP-IT directory for June 25, 2016 is "$ST_FPIT_AUX_DIR/2016/06/25". Within the MERRA-2 month directory, there are files for each day with filename format merra2_{year}{month}{day}.nc4. Within the FP-IT day of month directory, there are files for each 3-hour increment with filename format fpit_{year}{month}{day}.{hour}.nc4. For MERRA-2, all 8 of the 3-hour increments for a given day are stored in the same file. These different structures for MERRA-2 and FP-IT mirror the structure of the source data, where MERRA-2 is processed on a monthly basis whereas FP-IT is processed as each 3-hour increment becomes available.

The files are in NetCDF format, with a header file and a data file for each entry. The different atmospheric parameters (height, temperature, and specific humidity) are packaged in the same files. The files have the same format as the original files from NASA, except that only the parameters used by Single Channel Surface Temperature (height, temperature, and specific humidity, plus supporting values used to organize these parameters such as latitude, longitude, pressure level, and time) are kept.

4.7.1.5.2 Time Lag in Data Availability

The MERRA-2 product has a 20 to 50 day time lag before becoming available on its http site. The FP-IT product has an 18 hour time lag (often less in practice) before becoming available on its http site. As a result, The FP-IT product will be used for forward processing. Because the FP-IT product goes back only to January 1, 2000, MERRA-2 will be used for historical processing, as its product goes back to January 1, 1980. The local archive for MERRA-2 will go back only to the Landsat 4 launch date. Either MERRA-2 or FP-IT can be used in the date range where both are available.

The st_aux_merra_from_NASA_archive.py is appropriately run as a monthly script, since the MERRA-2 data is updated monthly. A cron job can be set up to run the script near the end of the month, requesting the date range to process as the entire previous month. The st_aux_fpit_from_NASA_archive script is appropriately run as a nightly script. Both scripts are flexible in that they can be run using date ranges.

4.7.1.5.3 Processing ASTER GED Dataset

A script is used to download tiles from the LP DAAC. The script uses a list of ASTER GED tiles as input, so updates to subsets of the ASTER GED can be made if needed. After downloading the tiles, the script extracts the data that the Single Channel Surface Temperature emissivity procedures require, which is emissivity mean, emissivity standard deviation, NDVI mean, latitude, and longitude. This is done using the h5copy tool. Then h5repack is used to reduce the size of the resulting file, and that reduced output is put in the local ASTER GED archive customized for Single Channel Surface Temperature use.

4.7.1.5.4 Time Lag in Data Availability

The ASTER GED is a static dataset, so there is no time lag in data availability. Preprocessing of the ASTER GED is done once. However, it would need to be repeated if the source dataset is refreshed.

4.7.1.5.5 Processing Elevation Dataset

If the elevation dataset used in level 1 processing is available, it is simply reformatted into the raw binary format required by the level 2 software. If it is not available, the elevation dataset can be constructed from the source GLS, GTOPO30, or RAMP.

4.7.2 LaSRC TOA Reflectance Algorithm

4.7.2.1 Background/Introduction

The Land Surface Reflectance Code (LaSRC) Algorithm generates Top of Atmosphere Reflectance (TOA-R) products. Processing of Operational Land Imager (OLI) reflectance band data results in an estimated in-band product that is TOA reflectance (unitless). This conversion occurs after Level 1 Radiometric (L1R) calibrations are applied.

4.7.2.2 Inputs

Description	Level	Source	Type
Scene (bias corrected, linearized)	N_{band} x N_{pixel}	Level 1 input data	Integer
TOA reflectance gain	N_{bands}	Metadata	Float
TOA reflectance bias	N_{bands}	Metadata	Float
Per-Pixel solar zenith angles	N_{pixel}	Level 1 Angle bands	Float

4.7.2.3 Outputs

Description	Level	Target	Type
Level 1 TOA reflectance	N_{band} x N_{pixel}		Integer

4.7.2.4 Algorithm Flow Diagram

Figure 4-108 shows the flow diagram of the LaSRC algorithm.

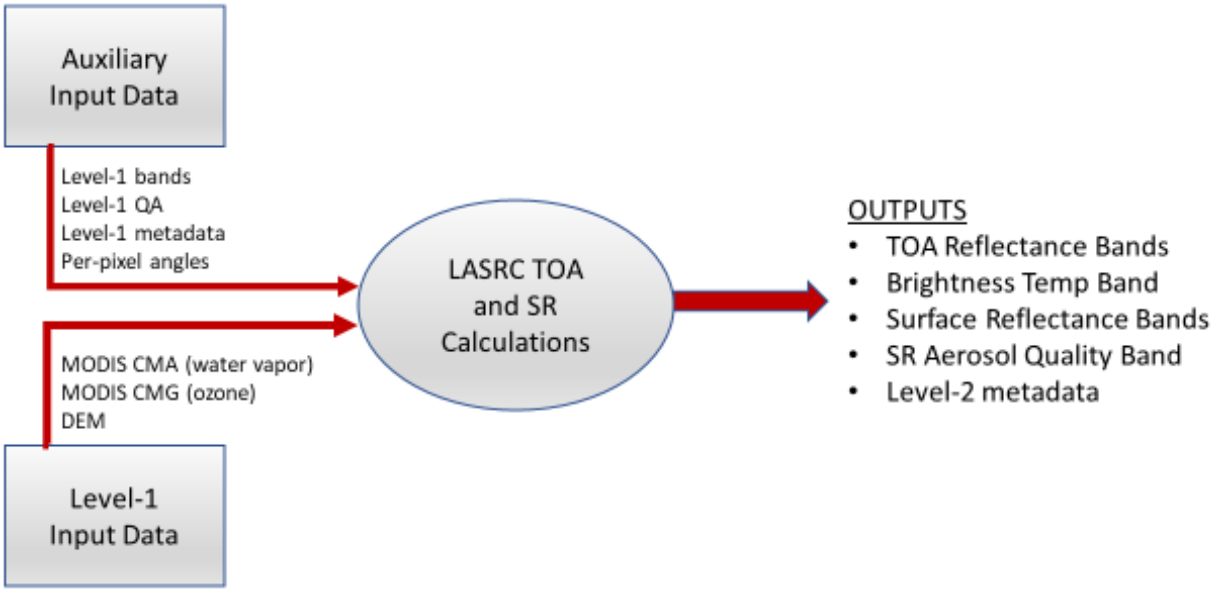

Figure 4-108. LaSRC Algorithm Flow Diagram

4.7.2.5 Procedure

TOA reflectance is calculated from gain and bias parameters in the metadata:

Ref(p) = (DN(p) * refl_gain + refl_bias) / cos(sza)

where Ref(p) = Output reflectance value for pixel ‘p’.
DN(p) = Input DN value (linearized and bias corrected) for pixel ‘p’.
cos(sza) = Cosine of the solar zenith angle. This value is calculated per-pixel from values in the metadata (scene solar angle and viewing geometry).
refl_gain = Reflectance gain, from metadata.
refl_bias = Reflectance bias, from metadata.

4.7.3 LaSRC TOA Brightness Temperature Algorithm

4.7.3.1 Background/Introduction

The Land Surface Reflectance Code (LaSRC) Algorithm generates Landsat 8/9 Top of Atmosphere Brightness Temperature (TOA-BT) products. Processing of Landsat 8/9 Thermal Infrared Sensor (TIRS) thermal data results in an estimated in-band product that is TOA-BT, expressed in the kelvin (K) unit of temperature. This conversion occurs after Level 1 radiometric (L1R) calibrations are applied.

4.7.3.2 Inputs

Description	Level	Source	Type
Scene (bias corrected, linearized)	N_{band} x N_{pixel}	Level 1 input data	Integer
TOA radiance gain	N_{band}	Metadata	Float

Description	Level	Source	Type
TOA radiance bias	N_{band}	Metadata	Float

4.7.3.3 Outputs

Description	Level	Type
Level 1 TOA thermal band product	N_{band} x N_{pixel}	Float

4.7.3.4 Procedure

To calculate brightness temperature, the thermal band data must first be converted to TOA radiance (W/m2-sr-mu). This is done using the radiance gain and bias from the metadata:

$$\mathrm{Rad(p)} = (\mathrm{DN(p)} * \mathrm{rad\_gain}) + \mathrm{rad\_bias}$$

where
Rad(p) = Output radiance value for pixel 'p'.
DN(p) = Input DN value (bias corrected) for pixel 'p'.
rad_gain = Radiance gain, from metadata.
rad_bias = Radiance bias, from metadata.

The brightness temperature is then calculated:

$$\mathrm{BT(p)} = \frac{\mathrm{K2}}{\log\left(1.0 + \left(\frac{\mathrm{K1}}{\mathrm{Rad(p)}}\right)\right)}$$

where
BT(p) = Output Brightness Temperature for pixel 'p'.
K2 = Thermal constant, value read from metadata.
K1 = Thermal constant, value read from metadata.

The constants K1 and K2 are dependent on the satellite instrument and the thermal spectral response of the band being processed. Note that target emissivity is normally part of this equation, but for brightness temperature the emissivity is assumed to be 1.

4.7.4 LaSRC Surface Reflectance Algorithm

4.7.4.1 Background/Introduction

The Land Surface Reflectance Code (LaSRC) Algorithm generates atmospherically corrected Surface Reflectance (SR) products. Processing of OLI reflectance band data results first in an estimated in-band product that is TOA reflectance (unitless). After that, the atmosphere must be taken into account. Processing of TOA reflectance products with an atmospheric retrieval algorithm results in an estimated in-band product that is Surface Reflectance (unitless).

4.7.4.2 Inputs

Description	Level	Source	Type	Variable
Level 1 QA	N_{pixel}	Level 1 input data	Integer	qaband

Description	Level	Source	Type	Variable
Level 1 TOA Reflectance	N_{band} x N_{pixel}	Level 1 input data	Scaled Integer	sband
Solar Zenith Angle at Scene Center		Metadata	Float	xts
Observation Zenith Angle at Scene Center		Metadata	Float	xtv
Terrain Altitude	N_{pixel}(CMG)	DEM	Integer	pres
Ozone	N_{pixel}(CMG)	MODIS CMG data		uoz
Water Vapor	N_{pixel}(CMG)	MODIS CMA data		uwv

4.7.4.3 Outputs

Description	Level	Type
Level 2 Surface Reflectance	N_{band} x N_{pixel}	Scaled Integer

4.7.4.4 Procedure

4.7.4.4.1 Theoretical Background

In the case of a Lambertian surface, minimal atmospheric water absorption, and with minimal atmospheric Rayleigh scattering, the surface reflectance (SR) calculation is:

$$\rho_s = \frac{\left[\frac{\rho_{TOA}}{tgOG * tgO3} - (\rho_{atm} - Xro_{rayp}) * tgwv_{half} - Xro_{rayp}\right]}{tr_{atm} * tgwv_{full} * (1 + S_{atm} * \rho_{partial})} \quad (1)$$

where ρ_s = Surface Reflectance (SR). Code variable is 'roslamb'.
$\rho_{partial}$ = Partial surface reflectance, described below.
ρ_{TOA} = Top of Atmosphere Reflectance (TOA-R). Code variable is 'rotoa'.
tg_{OG} = Gaseous transmission of other gasses. Code variable is 'tgog'.
tg_{O3} = Gaseous transmission of Ozone. Code variable is 'tgoz', often combined with other gasses as 'tgo = tgog * tgoz'.
ρ_{atm} = Intrinsic atmospheric reflectance. Code variable is 'roatm'.
$tgwv_{full}$ = Gaseous transmission of the full column of atmospheric water vapor. Code variable is 'tgwv'.
$tgwv_{half}$ = Gaseous transmission of half the column of atmospheric water vapor. Code variable is 'tgwvhalf'.
Xro_{rayp} = Rayleigh scattering reflectance. Code variable is 'xrorayp'.
tr_{atm} = Total atmospheric transmission. Code variable is 'ttatm'.
S_{atm} = Atmospheric spherical albedo. Code variable is 'satm'.

The 'partial surface reflectance,' $\rho_{partial}$, is an approximation of the total surface reflectance that is used as an aid in computation:

$$\rho_{partial} = \frac{\left[\frac{\rho_{TOA}}{tgOG * tgO3} - (\rho_{atm} - Xro_{rayp}) * tgwv_{half} - Xro_{rayp}\right]}{tr_{atm} * tgwv_{full}} \quad (2)$$

This approximation is valid when S_{atm} is near zero, which is generally the case for nadir-viewing sensors such as Landsat. Without this approximation the formula for ρ_s becomes much more complex and computationally expensive.

The above equation is used several times throughout the LaSRC SR calculation. It is performed in the subroutine function 'atmcorlamb2'. Many of the parameters in the atmosphere correction equation are calculated in atmcorlamb2 from scene metadata and then delivered as outputs for further use.

4.7.4.4.2 Perform the Lambertian atmospheric correction

The first step in LaSRC SR calculation is to calculate the basic parameters and perform a Lambertian-only atmospheric correction. This is done assuming that the water vapor transmittance (tgwv) is one and the Raleigh scattering coefficient (xrorayp) is zero.
For each band:

1. Save the input TOA reflectance values, as they will be needed for later corrections.
2. Create the basic parameters by calling atmcorlamb2 with some dummy input parameters. The input TOA reflectance (ρ_{TOA} or rotoa) is set to 0.0, and the Aerosol Optical Thickness parameter (raot550nm) is a dummy value meant to simulate a clear, dry atmosphere, currently 0.05.
3. Discard most of the outputs of atmcorlamb2, including output reflectance (roslamb), as they are not valid given the dummy inputs.
4. Save the valid output values: other gas transmittance (tg_{OG} or tgo), intrinsic atmospheric reflectance (ρ_{atm} or roatm), total atmospheric transmission (tr_{atm} or ttatmg), and atmospheric spherical albedo (s_{atm} or satm).
5. Perform the initial Lambertian atmospheric correction by using a simplified version of the surface reflectance formula with only those parameters:

$$\rho_{s1} = \frac{\left[\frac{\rho_{TOA}}{tgOG} - \rho_{atm}\right]}{tr_{atm} * (1 + S_{atm} * \rho_{partial1})} \tag{3}$$

where $\rho_{partial1}$ = A partial version of this initial correction, used to simplify the calculations:

$$\rho_{partial1} = \frac{\left[\frac{\rho_{TOA}}{tgOG} - \rho_{atm}\right]}{tr_{atm}} \tag{4}$$

6. Save this value as the preliminary corrected SR for this band.

4.7.4.4.3 Retrieve the atmospheric correction parameters

The next step in SR processing is to create a lookup table for atmospheric correction parameters for various values of Aerosol Optical Thickness (AOT).

1. Create a lookup table of useful AOT values (code variable is 'aot550nm').

For each band:

2. For each value in the AOT lookup table:
    a. Create correction parameters by calling atmcorlamb2 with each AOT value in the lookup table, but with a dummy value for the input TOA reflectance (ρ_{TOA} or rotoa) set to 0.0.
    b. Save the output values of Intrinsic atmospheric reflectance (ρatm or roatm), total atmospheric transmission (tratm or ttatmg), and atmospheric spherical albedo (S_{atm} or satm).
3. Save just one output value of other gas transmittance (tg_{OG} or tgo) and the Rayleigh scattering coefficient (xrorayp), as they are not dependent on the AOT.
4. With the correction parameters found for discrete values of the AOT, perform a cubic polynomial fit to create a continuous curve that describes roatm, ttatmg, and satm. These coefficient functions will later be used to provide values for these coefficients given any AOT input value.

4.7.4.4.4 Interpolate auxiliary data

Some input data, such as Water Vapor (uwv) and Ozone (uoz), are delivered at coarser resolution than Landsat data, and they need to be interpolated so that accurate values can be calculated for each pixel in the Landsat scene. Surface pressure (pres) is also calculated here, as it is derived from the coarse resolution digital elevation model (DEM) input data which must also be interpolated. Interpolation is not performed for any pixels marked as fill.

4.7.4.4.5 Aerosol inversion

Now that accurate models have been created for the correction parameters, those models are used to determine the correct AOT value to be used in the final SR calculation.

For each pixel in the scene:

1. Define a window (currently hardcoded as 3x3 pixels) around the current pixel.
2. Discard all pixels in this window that are fill.
3. Get the Normalized Difference Water Index (NDWI) for the current pixel.
    a. Find the correct interpolated location in the CMG ratio file, which is derived from MODIS data.
    b. The CMG-sourced ratios can give an estimate of NDWI, which is then used to calculate provisional values for band 1 (CA), 2 (Blue), and 7 (SWIR2).

        For each of these provisional bands:
        i. Iteratively call atmcorlamb2 with the provisional band values. This is performed over every value in the AOT lookup table and over three values for the Aerosol Extinction Coefficient (AEC, code variable ‘eps’). That gives a linear fit for AEC from which an optimized value can be obtained for both AEC and AOT. Save the AOT and AEC values derived for this pixel.

c. Test the optimized parameters by using them to call atmcorlamb2 with band 4 (Red) and band 5 (NIR) data. This is used to calculate a provisional Normalized Difference Vegetation Index (NDVI).
d. If the provisional NDVI is greater than zero (and the NIR band is nonzero), then this pixel appears to be clear land data with a valid aerosol retrieval.
e. If the provisional NDVI is less than zero, then this pixel appears to be water and is flagged as such.
    i. Retrieve the aerosol over water using band 4 (Red).
    ii. Call atmcorlamb2 with band 1.
    iii. Store the AOT and AEC values. Test the residual to validate this as a water pixel. If it is, mask the pixel as water and a clear aerosol retrieval. If it isn't a valid water pixel, mask it as not clear so that the stored AOT and AEC values are later replaced by the following filling process.

A series of passes through the image fills in the pixels with invalid aerosol inversion. At this time, only the center of the 3x3 windows will have aerosol values. The first loop looks at a 15x15 area around the pixel and computes the average of the valid aerosols around the pixel. That average is used to fill the current invalid pixel if there are at least four valid aerosols in the surrounding 15x15 window. The second loop uses less stringent criteria. This loop looks at the same 15x15 window around each invalid aerosol; however, it uses the average of the valid aerosols if there is at least one valid aerosol, including ones filled in this or the previous loop. The first two loops traverse through the image from top to bottom and left to right. After they are complete, if there are any remaining invalid pixels, a final loop traverses the image from bottom to top and right to left. This loop uses the average of the valid aerosols if there is at least one valid aerosol, including ones filled in the previous loops. Any remaining invalid pixels are set to use a default AOT of 0.05 and AEC of 1.5.

Now that each center pixel of the 3x3 aerosol window has a valid or filled aerosol value, the remaining pixels in each 3x3 aerosol window are interpolated using the representative center window values. This provides an aerosol value for every pixel in the scene, which are then used for further atmospheric correction.

4.7.4.4.6 Second atmospheric correction using the aerosols

As a final step, the parameters derived in earlier steps are used in one final atmospheric correction.

For each pixel in the scene:

1. Apply the Lambertian correction using equation [3] and the parameters derived in the Lambertian atmospheric correction step.
2. Call atmcorlamb2 to apply the final correction using equation [1] and the parameters derived in the retrieval of the atmospheric correction parameters and the aerosol inversion.
3. Scale and save the final surface reflectance value for this pixel.

4.7.5 Single Channel Landsat Surface Temperature Algorithm

The Single Channel Landsat Surface Temperature (ST) Level-2 algorithm produces image files that represent surface temperature in Kelvin (K) for each pixel in a Landsat scene. This algorithm was created at the Rochester Institute of Technology (RIT) and the NASA Jet Propulsion Laboratory (JPL) in cooperation with USGS software engineers.

4.7.5.1 Inputs

Description	Purpose
TOA Reflectance Bands	Multiple bands are used with other inputs to simulate a Landsat emissivity band matching the Landsat scene
Level-1 Thermal Band	Used to create a thermal radiance band
Quality Band	Used to identify cloud locations when generating distance to cloud band
Elevation Band	Used to adjust for the correction height during the interpolation step
MERRA-2 Data (Geopotential height, temperature, and specific humidity)	Provides a 0.625° x 0.5° grid of atmospheric data at 42 pressure levels. Used to generate inputs to MODTRAN, which is used to calculate atmospheric transmission, upwelled radiance, and downwelled radiance at the grid points at multiple elevations (Mesinger, DiMego, Kalnay, Mitchell, & Shafran, 2006).
FP-IT Data (Geopotential height, temperature, and specific humidity)	Provides a 0.625° x 0.5° grid of atmospheric data at 42 pressure levels. Used to generate inputs to MODTRAN, which is used to calculate atmospheric transmission, upwelled radiance, and downwelled radiance at the grid points at multiple elevations (Lucchesi, 2013)
Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER) Global Emissivity Dataset (GED) Emissivity Band	Used to simulate a Landsat emissivity band matching the Landsat scene.
ASTER GED Emissivity Standard Deviation	Used to calculate emissivity uncertainty which is a factor in the overall ST uncertainty/QA band.
ASTER GED Normalized Difference Vegetation Index (NDVI) Bands	Used to simulate a Landsat emissivity band matching the Landsat scene.

4.7.5.2 Outputs

Description	Units	Type
Surface Temperature	Kelvin (K)	16-bit Integer
Surface Temperature Quality (Uncertainty)	Kelvin (K)	16-bit Integer
Emissivity (Intermediate)		16-bit Integer
Emissivity Standard Deviation (intermediate)		16-bit Integer
Thermal Radiance (Intermediate)	Radiance ($W\ m^{-2}\ sr^{-1}\ mu^{-1}$)	16-bit Integer
Atmospheric Transmission (Intermediate)		16-bit Integer
Upwelled Radiance (Intermediate)	Radiance ($W\ m^{-2}\ sr^{-1}\ mu^{-1}$)	16-bit Integer

Downwelled Radiance (Intermediate)	Radiance ($W\ m^{-2}\ sr^{-1}\ mu^{-1}$)	16-bit Integer
Distance to Cloud (Intermediate)	Distance (km)	16-bit Integer

4.7.5.3 Algorithm Flow Diagram

Figure 4-109 shows the flow diagram of the Single Channel Landsat Surface Temperature (ST) product.

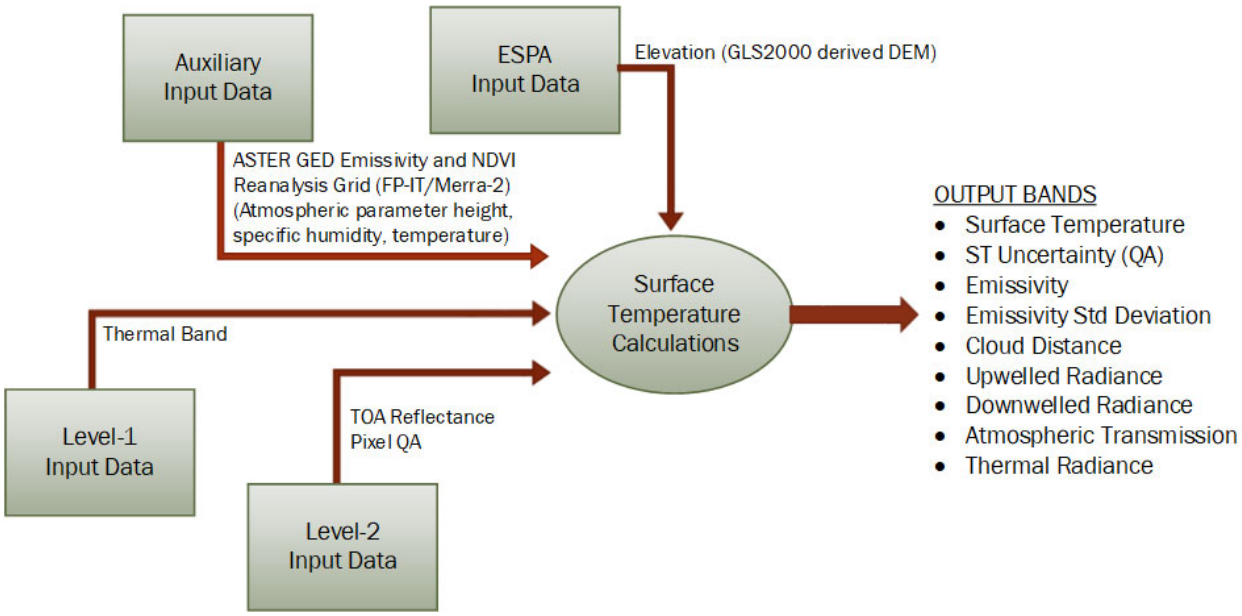

Figure 4-109. Single Channel Landsat Surface Temperature Algorithm Flow Diagram

4.7.5.4 Procedure

The ST algorithm produces image files representing surface temperature (K) for each pixel in a Landsat scene.

The algorithm is used to create Landsat ST products with single band methods using external data sources for atmospheric parameters and requiring emissivity.

The ST algorithm uses parameters from the MERRA-2 and GEOS-5 FP-IT datasets as auxiliary input. These are global datasets, so the algorithm is applicable worldwide (within the constraints of other inputs to the algorithm).

The ST algorithm involves the following steps:

1. Determine reanalysis (MERRA-2 or FP-IT) grid points for the scene
2. Extract auxiliary reanalysis data
3. Build MODerate resolution atmospheric TRANsmission (MODTRAN) software input
4. Create emissivity band
5. Create emissivity standard deviation band

6. Run MODTRAN
7. Create atmospheric parameter bands
8. Build surface temperature band
9. Create distance to cloud band
10. Create ST quality assessment (QA) band

4.7.5.4.1 Determine Reanalysis Grid Points

This procedure builds a grid of points based on the data coverage of both the reanalysis (MERRA-2 or FP-IT) data and Landsat scene, plus a buffer. The buffer allows points outside the scene to be used, as they are needed during later interpolation steps for pixels near the scene edges.

The TOA-BT band is used to define the size and projection information for the Landsat scene. Landsat metadata is used to define the bounding coordinates. These coordinates are converted to x and y meters, adjusted to the MERRA-2 and FP-IT data resolution (0.625º x 0.5º), and converted back to latitude and longitude.

A static file with the reanalysis row, reanalysis column, latitude, and longitude for all 361 x 576 MERRA-2 or FP-IT points is read, and the minimum and maximum rows and columns from that file that overlap the adjusted Landsat scene boundaries are calculated. The number of rows and columns in the grid are calculated from the minimum and maximum values.

A list of grid points within the range is created with information on reanalysis row and column, grid row and column, and point location for each point.

The grid point list needs to track whether the point should have MODTRAN run. To determine this, the Landsat scene is checked for valid data. An initial check is made for points inside the Landsat scene boundaries. Those points are set to have MODTRAN run. Then the edges of valid scene data are found.

For all pixels on the edges, the closest grid point and the surrounding 8 grid points are found, followed by the distances of the grid points to the pixel. The quadrant (center grid point and 3 other grid points out of the 9 grid points) with the lowest average distance to the pixel is determined, and the grid points in that quadrant are marked to have MODTRAN run. Some of the points may be outside of the scene boundaries, but they are needed during interpolation in later phases.

The grid point list is written to a binary file. Header information (number of grid points, number of grid rows and columns) is written to an American Standard Code for Information Interchange (ASCII) file.

Input	Output
Top of atmosphere brightness temperature (BT) band	Grid point file
Landsat scene metadata	Grid point header file
MERRA-2 or FP-IT coordinates file in ST static data dictionary	MODTRAN “to run” and “to not run” grid point files

Table 4-64. Reanalysis Grid Point Determination Input and Output

4.7.5.4.2 Extract Auxiliary Reanalysis Data

This procedure obtains the acquisition date and scene center time from the Landsat scene metadata. MERRA-2 and FP-IT data is produced by NASA in 3-hour increments, and the data is available in those 3-hour increments for the parameters that ST requires (height, temperature, and specific humidity).

The procedure finds the reanalysis (MERRA-2 or FP-IT) data entries before and after the Landsat scene acquisition time, and then builds the path and filename to the auxiliary data for these dates and times for atmospheric parameters height, temperature, and specific humidity. It does the same for the output files.

Reanalysis data for 42 pressure levels for the 2 times and 3 atmospheric parameters are extracted using netCDF libraries. It is anticipated that there will be some gaps in MERRA-2 and FP-IT reanalysis data. In those cases, interpolation is used to fill in missing values with surrounding valid values, and extrapolation is used to fill in missing values beyond the valid values. After this the values are written to separate binary output files. The output files are organized in 6 separate directories for the 2 times and 3 parameters.

Input	Output
MERRA-2 or FP-IT auxiliary archive in NetCDF format filtered from original MERRA-2 or FP-IT dataset	Binary output files
Landsat scene metadata	

Table 4-65. Auxiliary Reanalysis Data Extraction Input and Output

4.7.5.4.3 Build MODTRAN Input

This step creates the input files used later by MODTRAN. These files are in top-level directories for each grid point, named by the grid row and column and the reanalysis (MERRA-2 or FP-IT) row and column. Within each are directories for the elevations to process, along with directories for the standard temperatures to process. Each directory also contains 1 albedo directory, which contains the MODTRAN input files.

The grid points file and standard atmosphere file are read. These values are required by MODTRAN for altitudes higher than the reanalysis data coverage.

MODTRAN template files for the head and tail of the tape5 file are then read. The tape5 file is the main input file that controls a MODTRAN run. This ASCII file has a specific

format historically derived from punch card use. The format can be conveniently divided into head, middle, and tail sections. The templates are close to the final required format, and only a few elements need to be updated for a specific ST MODTRAN run.

The reanalysis (MERRA-2 or FP-IT) data previously extracted for all grid points, pressure layers and parameters (temperature, specific humidity, and height) is read next. These values are put in a 2-dimensional structure organized by reanalysis rows and columns.

The procedure then determines what altitudes to process. A standard set of 11 elevations can be processed, and all of these elevations are needed for the highest mountain peaks. However, MODTRAN is run multiple times for each grid point and elevation, which takes a significant amount of time. It is desireable to avoid processing higher elevations for scenes that do not need it because they do not include land at those elevations. To do this, the elevation file is read. Then the minimum and maximum elevations in that file are determined and scaled from meters to kilometers to match the reanalysis data. The standard altitudes that bound the minimum and maximum elevations in the scene are then determined, and the list of elevations that need to be processed through MODTRAN is built. This list includes the 0 elevation, plus all the standard elevations from the one bounding the minimum to the one bounding the maximum. Therefore, a scene with low elevations could process as few as 2 elevations, whereas a scene with high elevations might have to process as many as 11 elevations. The list of elevations to process is written to the MODTRAN elevations file.

The following steps are performed for each point in the grid file that requires a MODTRAN run.

1. Reanalysis units are converted to units that MODTRAN can use. For each pressure layer:
    a. The geopotential height at the 2 reanalysis times surrounding the scene time is converted to geometric height at those times
    b. The specific humidity at those times is also converted to relative humidity. During this step, the reanalysis temperature is also extracted and used. The reanalysis temperature is already suitable for use by MODTRAN and does not need to be converted like the other reanalysis parameters.
    c. The geometric height, relative humidity, and temperature from reanalysis times 0 and 1 around the scene time are linearly interpolated to derive their values at the scene center time.
2. The set of elevations to process is defined to be the same as the set previously derived, except that the 0 elevation is the reanalysis height at the bottom pressure layer, or 0 if that value is negative. The 0 elevations are written to the grid elevations file in 2 formats. The first is written with more precision for science calculations, and the second is shorter to match the elevation directory name so the directory can be easily looked up later.

3. The final three steps are performed for each elevation to process. The purpose of these steps is to fill in template sections of the tape5 file used as input to each MODTRAN run with values appropriate for the elevation.
    a. The pressure layers for the elevation are determined. The pressure layers with heights that are immediately above and below the elevation are found. The pressure, temperature, and relative humidity for the elevation are estimated by interpolating those values from the heights above and below it. These values begin a list of base pressure layers but is done only if the elevation is not too close to the pressure layer height to avoid having MODTRAN process both values, since MODTRAN will fail if the values are too close.
    b. The height, temperature, and relative humidity of each pressure layer with height above the elevation, as well as the pressure level itself, are then added to the base pressure level list. Those values are derived from the reanalysis data.
    c. The maximum height of the base pressure layers is determined. Above the available reanalysis-derived heights, standard atmospheric values for height, temperature, relative humidity, and pressure are used. Interpolation between the reanalysis-derived values and standard atmospheric values may be used to smooth the transition between the 2 data sources, and then the standard atmospheric values are added to the list of pressure layers to process.
4. The tape5 head template is updated with the elevation, and the middle of the tape5 file is updated with height, pressure, temperature, and relative humidity for the list of pressure layers.
5. MODTRAN is set up to later be run for three standard pairs of temperature and albedo for each grid point and elevation to process. For each pair, the temperature and albedo in the header section of the tape5 file is updated, and the tape5 file (built from the head, body, and tail that have been constructed) is written so it can later control a MODTRAN run.

Input	Output
Grid point files	Directory structure for multiple MODTRAN runs
MODTRAN ASCII template files for head and tail of tape5 file in ST static data directory	MODTRAN tape5 input file for each run
Standard MODTRAN mid-latitude summer atmosphere ASCII file with height, pressure, temperature, and relative humidity in Surface Temperature static data directory	Grid elevations file with 0 elevations for each grid point
Elevation file	MODTRAN elevations file with the number of elevations to process, followed by the elevations
Landsat scene metadata	
Binary files created in Section 4.7.5.4.2	

Table 4-66. Build MODTRAN Input Phase Input and Output

4.7.5.4.4 Create Emissivity Band

This procedure creates emissivity data so it can later be used with Landsat thermal data.

This procedure retrieves Landsat metadata, and then selects pre-defined sensor coefficients that help with conversion between ASTER and Landsat emissivity, snow emissivity, and vegetation.

The procedure extracts the Landsat NIR and red bands, and computes Landsat NDVI. Similarly, it extracts the Landsat green and red bands, and computes Normalized Difference Snow Index (NDSI). Snow is defined to be pixel locations where NDSI is greater than 0.4.

The procedure then calculates an initial estimate of emissivity by adjusting the ASTER band 13 and 14 GED emissivity data so it can later be used with Landsat thermal data. It also builds an NDVI band from the ASTER NDVI dataset. This is done using the following steps:

1. The procedure loops through all latitude and longitude pairs covered by the ranges between Landsat scene corners, downloading the ASTER GED tiles for b13 emissivity, b14 emissivity, and NDVI at that location. For emissivity, it scales the b13 and b14 data, combines that data using the sensor regression coefficients to produce the estimated Landsat emissivity for the tile. Then it sets any locations that are below threshold 0.6 in the original b13 or b14 data to nodata, since those locations are considered to be invalid in the ASTER GED data (likely due to undetected clouds). For ASTER NDVI, it scales the data.

2. Once the tiles for the latitude and longitude pairs are created, they are mosaiced to cover the full Landsat scene. This is done once for the Landsat emissivity data, and once for the ASTER NDVI data.

3. The data is then warped to the Landsat scene projection and clipped to the Landsat scene extent. This is done once for the Landsat emissivity data, and once for the ASTER NDVI data. These outputs are saved to a file.

An initial estimate of emissivity based on ASTER data is now available for the Landsat scene. Since the ASTER GED dataset averages emissivity spanning years 2000 to 2008, these emissivity values need to be adjusted to estimate the emissivity for the current Landsat scene at a particular time.

The major factors that could change emissivity that are considered are vegetation and snow cover. To do this, the warped Landsat emissivity and ASTER NDVI data are first read. The minimum and maximum Landsat NDVI values are found, as well as the maximum ASTER NDVI value.

The Landsat and ASTER NDVI bands are scaled by their maximum values. Bare soil locations are defined to be locations where ASTER NDVI is less than 0.5. The Landsat emissivity value for these pixels is adjusted by the ASTER NDVI values at those locations. For the other pixels, the Landsat emissivity value is adjusted by the sensor vegetation coefficient and the Landsat NDVI data at those locations. The pixels that were determined to be snow are simply defined to have the sensor snow emissivity coefficient value. Landsat emissivity values that are greater than a water emissivity constant are adjusted to be the water constant. Landsat emissivity values that are less than threshold 0.6 are set to nodata, since this result is not considered physically realistic for Earth scenes. Finally, the quality band is read, and water locations are identified from that information. Any location identified as water in the quality band that is nodata in the emissivity band is assigned a standard water emissivity value in the emissivity band. This adjusted Landsat emissivity band is written to file, and a temporary band with the locations that were updated to the water emissivity value based on the quality band is also written.

Input	Output
ASTER GED bands 13 and 14 emissivity tiles	Emissivity band matching Landsat scene coverage
ASTER GED NDVI tiles	Emissivity nodata water mask
Landsat top of atmosphere reflectance bands (green, red, NIR, SWIR)	
Quality band identifying water locations	
Landsat metadata (Landsat top of atmosphere brightness temperature band is used for extent and projection)	

Table 4-67. Emissivity Band Creation Input and Output

4.7.5.4.5 Create Emissivity Standard Deviation Band

The procedure used to create the emissivity standard deviation band is similar to the one used to create the emissivity band but does not include many of the adjustments. This procedure was not combined with the emissivity mean and NDVI procedure to avoid increasing memory use in that procedure. However, the 2 procedures share several utility functions.

This procedure scans through the latitude and longitude pairs covered by the Landsat scene, downloading the ASTER emissivity standard deviation tile for bands 13 and 14 for that location. The ASTER data is scaled and combined using the square root of the squares of the bands' data divided by 2. The tiles are mosaicked and warped using a similar procedure to the one used for the ASTER emissivity mean and ASTER NDVI to build a band matching the Landsat extent. The mask for emissivity nodata locations that were updated to water emissivity is read, and masked locations are assigned an emissivity standard deviation value of 0.

Input	Output
ASTER GED bands 13 and 14 emissivity standard deviation tiles	Emissivity standard deviation band matching Landsat scene coverage
Emissivity nodata water mask	
Landsat scene metadata	

Table 4-68. Emissivity Standard Deviation Band Creation Input and Output

4.7.5.4.6 Run MODTRAN

This procedure runs MODTRAN for each grid point, elevation and temperature/albedo pair to be processed.

The MODTRAN input files are located in top-level directories for each grid point, named by the grid row and column and the reanalysis row and column. Below that are directories for the elevations to process. Below that are directories for the standard temperatures to process. There is one albedo directory under each of these, containing the MODTRAN input files.

The “Run MODTRAN” phase typically takes a long time to run. There will typically be a few dozen points to process in a Landsat scene when MERRA-2 or FP-IT input is used. Each point will have from 2 to 11 elevations, depending on the elevation variation in the scene. Some scenes are able to process much quicker than others because of this factor. Each elevation will have three temperature/albedo pairs to process. Therefore, several hundred MODTRAN runs are typically needed. This procedure supports several parallel MODTRAN runs if hardware is available for that. This is controlled using a configuration parameter.

The procedure starts by reading the grid points. These identify the point row, point column, reanalysis row, and reanalysis column for each grid point. They also identify which points should have MODTRAN run. For those points, the directory based on the grid and reanalysis rows and columns is built.

For each point, the subdirectory tree under the top-level directory is constructed. In each of these directories, MODTRAN is run using the input files built in previous sections. This creates files tape6 and pltout.asc.

MODTRAN produces many results, and the ST procedure needs only a small portion of those. The “AREA-AVERAGED GROUND TEMPERATURE [K]” value in the tape6 file is extracted and written to the st_modtran.hdr file. The pltout.asc results are slightly formatted and written to the st_modtran.data file. The length of that data is written to the st_modtran.hdr file.

Input	Output
MODTRAN input files for each grid point, elevation, and temperature/albedo pair to process Grid points file	st_modtran.hdr header file and st_modtran.data data file for each MODTRAN run

Table 4-69. MODTRAN Run Input and Output

4.7.5.4.7 Create Atmospheric Parameter Bands

This procedure creates bands for several atmospheric parameters. These parameters include atmospheric transmission, upwelled radiance, and downwelled radiance. It also creates a thermal radiance band based on the Landsat thermal band.

The atmospheric parameter band step starts by reading several files. First, the grid point header file is read to get the grid point count as well as the number of grid rows and columns. It then reads the grid points into a grid_points structure, and the number of elevations and the elevation values from the modtran_elevations.txt file.

For each grid point, the grid point values are copied into the modtran_points structure. The elevations for the structure are allocated based on the number of elevations actually processed, and the elevation values are added to the structure.

Finally, for the MODTRAN points that were run, the grid elevations file is read, and the values are put in the modtran_points structure at the 0 elevation. There is a short version of each 0 elevation for the elevation directory name, and a long version for science calculations.

Atmospheric parameters transmission, upwelled radiance, and downwelled radiance are then calculated for each MODTRAN point. This process starts with the sensor's spectral response file, which is a predefined file that contains rows with wavelength and spectral response at that wavelength. The blackbody radiance is then calculated for temperatures 273K and 310K, two of the temperatures with MODTRAN results. The blackbody radiance is calculated at each wavelength using Planck's equation and the spectral response.

The remaining steps to compute the three atmospheric parameters at each MODTRAN point are done for each grid point at each elevation for the points where MODTRAN was run.

For each of these, the 0K temperature/0.1 albedo pair's zero elevation temperature (the "0K" temperature MODTRAN run actually used the air temperature of the initial atmospheric layer, not literally "0K") and record count are read from the MODTRAN header file. The record count is the same for all three temperatures (0K, 273K, 310K), so that value is used for all of them. For all three temperatures, the wavelength and corresponding radiance is read from the st_modtran.data file. These values are organized in an array with four columns representing wavelength, "273K/0 albedo" radiance, "310K/0 albedo" radiance, and "0K/0.1 albedo" radiance.

The observed radiances from the MODTRAN results are calculated, first for the 273K MODTRAN inputs (column index 1), and then for the 310K MODTRAN inputs (column index 2). These calculations are similar to those for the blackbody radiance, but instead of using Planck's equation they interpolate over the MODTRAN results. Using these results, transmittance and upwelled radiance are calculated.

Downwelled radiance is the last to be derived. The blackbody radiance and observed radiance are calculated, this time for 0K (column index 3). Downwelled radiance is then calculated as ((observed radiance at 0K– upwelled radiance) / transmittance) – (blackbody radiance at 0K * water emissivity)) * inverse water albedo.

Finally, transmittance, upwelled radiance, and downwelled radiance are saved in the modtran_points structure for the MODTRAN point and elevation.

When all MODTRAN points are processed, the latitude, longitude, elevation, transmittance, upwelled radiance, and downwelled radiance for each elevation for each point are written to ASCII file atmospheric_parameters.txt. This file can be used for analysis.

While transmittance, upwelled radiance, and downwelled radiance have now been calculated for each MODTRAN point, these parameters, along with the thermal radiance for all pixels in the Landsat scene are required. To obtain this, the Landsat thermal band is read and converted to radiance using the Landsat metadata gain and bias values. The elevation band is then read.

The remaining steps to build the four bands are done for each valid pixel in the Landsat scene:

1. For each pixel, the longitude, latitude, easting, and northing values are found.

2. The grid point that is closest to the pixel is determined (aka the center grid point). From the center point, the surrounding eight points (lower left, lower center, etc.) are identified, and the distances from those points to the pixel are determined. The nine points are in a 3x3 layout.

3. There are four quadrants in the layout, each with four points. The average distances of the quadrants' points to the pixel are found. The center point is ignored in this calculation because all four quadrants share it. The quadrant with the lowest average distance to the pixel is chosen as the closest quadrant.

4. The elevation band height at the pixel location is converted from meters to km to match the MODTRAN elevations. Using the modtran_points structure for the four grid points in the chosen quadrant, the elevations above and below the pixel's elevation that have MODTRAN results are then determined. The atmospheric transmittance, upwelled radiance, and downwelled radiance in the MODTRAN results are interpolated at these elevations to their values at the pixel's elevation. At this step the three parameters are known at the pixel's elevation at the four nearby grid points. The three parameters at those four locations are interpolated to the location of the pixel, giving weights for each grid point based on its distance to the pixel.

When the atmospheric parameters are calculated for each pixel, the three parameters and thermal radiance are written to band files.

Input	Output
Landsat thermal band	Thermal radiance band
Elevation band	Atmospheric transmission band
Spectral response file for sensor	Upwelled radiance band
Environment variable ST_DATA_DIR pointing to spectral response file location	Downwelled radiance band
Grid point file	Atmospheric parameters ASCII file
Grid point header file	
MODTRAN elevations file	
MODTRAN results data file and header file from each MODTRAN run	
Grid elevations file	
Landsat metadata	

Table 4-70. Atmospheric Parameter Band Creation Input and Output

4.7.5.4.8 Build Surface Temperature Band

This procedure builds the main product, the surface temperature band, using the atmospheric parameter bands, the thermal radiance band, and the emissivity band created in earlier steps. It also uses a brightness temperature lookup table for the sensor being processed.

The input bands (thermal radiance, atmospheric transmission, upwelled radiance, downwelled radiance, and emissivity) are read, and the following calculations are done on a band-wide basis, with handling for no data locations:

1. Surface radiance is calculated (thermal radiance – upwelled radiance) / atmospheric transmittance.
2. Radiance is calculated as surface radiance – (1.0 – emissivity) * downwelled radiance. Planck emitted radiance is then radiance / emissivity.
3. The brightness temperature lookup table for the sensor is read. This lookup table contains temperature values from 150K to 373K in 0.01K increments, covering the full range of Earth temperature values typically expected (volcanoes and wildfires may exceed this), and the corresponding radiance value for the sensor's spectral response. The Earth-emitted radiance value is interpolated using this table to derive the surface temperature.
4. The floating-point surface temperature is then scaled, converted to 16-bit unsigned integer, and written as a band to disk.

Input	Output
Thermal radiance band	Surface temperature band – UINT16
Atmospheric transmission band	
Upwelled radiance band	
Downwelled radiance band	
Emissivity band	
Brightness temperature lookup table for the sensor	
Environment variable ST_DATA_DIR pointing to lookup table location	

Table 4-71. ST Band Building Input and Output

4.7.5.4.9 Build Distance to Cloud Band

This procedure creates a band where each pixel contains the distance, in kilometers, that pixel is from the nearest cloud pixel marked in the quality band.

The distance to cloud procedure first reads the quality band, and then finds the locations of clouds using that band and creates a boolean band in memory identifying the cloud locations. It then performs a distance transformation using the cloud location information. The transformation creates a band of distances to nearest cloud in pixels. This distance band is converted to kilometers.

The distance transformation operates on all pixels, so locations identified as fill in the quality band are set to fill in the distance to cloud band, and that band is written to disk.

Input	Output
Quality band with cloud locations	Distance to cloud band – 32-bit float

Table 4-72. Distance to Cloud Band Building Input and Output

4.7.5.4.10 Build Surface Temperature Quality Band

The surface temperature quality band procedure builds a band with estimates of ST uncertainty in K units, by considering the contributions to uncertainty from the atmosphere, the instrument, emissivity, cross correlation of these values, and unknown uncertainty.

The input bands thermal radiance (L_{th}), atmospheric transmission (τ), upwelled radiance (L_{uw}), downwelled radiance (L_{dw}), emissivity (ε), emissivity standard deviation (ε_{sd}), and cloud distance (d) are read. Fill locations are extracted to calculate valid data only and are restored at the end of the procedure. Many of the calculations in this procedure are done at a band level.

A radiance image (L) is built to provide nominal values around which uncertainty values will operate using

$$L = \frac{L_{th} - L_{uw} - (1 - \varepsilon)L_{dw}\tau}{\varepsilon\tau}$$

To calculate the atmosphere's contribution to the ST product's uncertainty, partials are calculated for atmospheric transmission, upwelled radiance, and downwelled radiance, with uncertainty calculated for these parameters using their respective bands. These calculations all use coefficients that are predefined based on simulations. The uncertainty values for the three parameters and their partials are then combined.

To calculate instrument uncertainty in radiance units, predefined instrument uncertainty values in radiance units – based on instrument uncertainty values in Kelvin units – are used. These are combined with a partial for thermal radiance.

To calculate emissivity uncertainty (u_ε), first an emissivity partial ($\frac{\partial L}{\partial \varepsilon}$) is calculated for emissivity as:

$$\frac{\partial L}{\partial \varepsilon} = \frac{L_{uw} - L_{th} + L_{dw}\tau}{\tau \varepsilon^2}$$

Then, predefined root mean square error (RMSE) values for each instrument derived during the emissivity investigations (ε_{RMSE}) are used, along with previously derived ASTER GED emissivity uncertainty values for bands 13 and 14. The uncertainty from the ASTER bands is combined into a single value (ε_{ASTER}). The overall emissivity contribution to uncertainty combines these predefined values and the emissivity partial with the emissivity standard deviation band using:

$$u = \left(\frac{\partial L}{\partial \varepsilon}\sqrt{\varepsilon_{sd}^2 + \varepsilon_{sd}^2 + \varepsilon_{sd}^2}\right)^2$$

.

The cross-correlation term is calculated using atmospheric transmission, upwelled radiance, downwelled radiance, and the related partials, again using coefficients that are predefined based on simulations.

Unknown uncertainty is derived based on cloud distances and atmospheric transmission. A predefined “unknown error matrix” and predefined center values for cloud distance and atmospheric transmission at different bins are also used. Unknown uncertainty is displayed in K, which is required for the final uncertainty band. The other values, however, are in radiance units, so the unknown uncertainty is converted to radiance to match them.

The nominal radiance band is converted to a temperature band with a version of the inverse Planck function that uses K1 and K2 constants for the sensor from the Landsat metadata. The unknown uncertainty in temperature units is divided by 2 to give a temperature delta band. This value is used to build temperature bands above and below the nominal temperature band, so the difference between these bands is the unknown uncertainty in temperature units. The temperature bands above and below the nominal band are converted to radiance with a version of the Planck function that uses the K1 and K2 constants. The total unknown uncertainty in radiance units is the differences between the two radiance bands.

Now that unknown uncertainty is converted from temperature to radiance units, it can be combined with the other uncertainty factors. The total surface temperature uncertainty in

radiance is the square root of the sum of the various uncertainty factors (atmosphere, instrument, emissivity, cross correlation terms, unknown uncertainty).

The uncertainty value is then converted from radiance to K. However, the Planck conversion functions cannot be used with uncertainty values; instead, they need to be uncertainty values around a nominal radiance image. That image is already available in memory, but calculations which divide the uncertainty in radiance by 2 to get delta radiance and use delta radiance above and below the nominal radiance image are required.

The radiance images above and below the nominal radiance image are converted to temperature using the inverse Planck function. The overall surface temperature uncertainty in K is the difference between the converted bands. This is written to disk as the surface temperature quality/uncertainty band.

Input	Output
Thermal radiance band	Surface temperature quality band with uncertainty in K – INT16
Atmospheric transmission band	
Upwelled radiance band	
Downwelled radiance band	
Emissivity band	
Emissivity standard deviation	
Cloud distance Band	
Landsat metadata	

Table 4-73. Input and Output for Building ST Quality Band

4.7.5.4.11 Convert and Scale Intermediate Bands

This procedure converts the intermediate bands (Thermal Radiance, Atmospheric Transmission, Upwelled Radiance, Downwelled Radiance, Emissivity, Emissivity Standard Deviation, and Cloud Distance) from a 32-bit floating point number to a 16-bit integer and scales them. Finally, the metadata within Landsat products needs to be updated with the data type, range, and scale factor.

4.7.6 Level 2 Pseudo-Invariant Calibration Sites (PICS) Characterization

4.7.6.1 Background/Introduction

This algorithm calculates basic statistics from geographic regions of interest (ROI) extracted from Level 2 Landsat products. The ROIs can be defined as polygons with vertices given as latitude/longitude coordinates or by a ROI center latitude/longitude coordinates and a width and height of the region. The algorithm can be applied to any site and, in combination with in-situ surface measurements, used for Level 2 surface reflectance and surface temperature validations. If the algorithm is used to repeatedly characterize uniform and stable radiometric pseudo-invariant calibration sites (PICS), the results saved in the database provide means for an automatic monitoring of temporal stability of OLI and TIRS instruments.

4.7.6.2 Inputs

Description	Symbol	Units	Level	Source	Type
Level 2 product (scaled surface reflectance / surface temperature), including the Quality band	Q_{cal}	*DN*	L2	L2 product	Unsigned 16-bit int
Surface reflectance multiplicative scaling factor	M_ρ	DN^{-1}	$N_{reflective\_band}$	Metadata	Double
Surface reflectance additive scaling factor	A_ρ	*unitless*	$N_{reflective\_band}$	Metadata	Double
Surface temperature multiplicative scaling factor	M_T	K/DN	1	Metadata	Double
Surface temperature additive scaling factor	A_T	K	1	Metadata	Double
ROI corner or center coordinates (lat/lon)		decimal degrees	$N_{corners}$ x N_{ROI} x 2	Shape file	Double
ROI width and height (if ROI is defined by the center coordinates)		pixels	N_{ROI} x 2	Shape file	Double
Line-Of-Sight model					
Geometric grid					
Digital Elevation Model					

ROIs are defined using scene specific shape files. The shape files are ASCII files containing lists of latitude/ longitude coordinates that represent the vertices of polygons that shape defined ROIs. The regions can also be described by a center latitude/ longitude and a width and height (in pixels), as region 2 in the figure below. Multiple ROIs, separated by blank lines, may exist in a single shape file and the two types of regions can be mixed. Shape file names have the form: wrs2_<path>_<row>.shp), where <path> and <row> specify the 3 digit path and row values which contain the regions defined by the file. The WRS association is required by the IAS for automatically running the PICS algorithm for specified sites, but the algorithm may be run using a shape file with any name, if so desired. There are no naming requirements for the region identifiers, but it is recommended to include user information, such as initials and/or the affiliated institution designation.

```
Filename

region_1_identifier
latitude_1 longitude_1
latitude_2 longitude_2
...
latitude_n longitude_n

region_2_identifier
center_latitude center_longitude
width height

...

region_N_identifier
latitude_1 longitude_1
latitude_2 longitude_2
...
latitude_n longitude_n
```

Figure 4-110. Shape File Description

4.7.6.3 Outputs

Description	Symbol	Units	Level	Target	Type
Surface reflectance minimum	ρ_{min}		$N_{reflective\_band}$ x N_{ROI}	Db / Report	Double
Surface reflectance maximum	ρ_{max}		$N_{reflective\_band}$ x N_{ROI}	Db / Report	Double
Surface reflectance mean	$\bar{\rho}$		$N_{reflective\_band}$ x N_{ROI}	Db / Report	Double
Surface reflectance standard deviation	σ_{ρ}		$N_{reflective\_band}$ x N_{ROI}	Db / Report	Double
Surface temperature minimum	T_{min}	K	N_{ROI}	Db / Report	Double
Surface temperature maximum	T_{max}	K	N_{ROI}	Db / Report	Double
Surface temperature mean	$\bar{T}$	K	N_{ROI}	Db / Report	Double
Surface temperature standard deviation	σ_T	K	N_{ROI}	Db / Report	Double
Coefficient of variation	C_v		N_{band} x N_{ROI}	Report	Double
Sun azimuth	φ	degrees	N_{band} x N_{ROI}	Db / Report	Double
Sun elevation	θ_{SE}	degrees	N_{band} x N_{ROI}	Db / Report	Double
Earth-Sun distance	d	AU	N_{band} x N_{ROI}	Db / Report	Double
Total number of pixels in ROI	N_{Total}	Pixels	N_{band} x N_{ROI}	Db / Report	Int
Valid pixels percentage	N_{Valid}	%	N_{band} x N_{ROI}	Db / Report	Int
Fill pixels percentage	N_{Fill}	%	N_{band} x N_{ROI}	Db / Report	Int
Cloud cover percentage	N_{CC}	%	N_{band} x N_{ROI}	Db / Report	Int
ROI identifier			N_{band} x N_{ROI}	Db / Report	String

In addition to the characterization parameters above, the algorithm generates and outputs thumbnail images of ROIs for each band, in ENVI format.

4.7.6.4 Options

Typically, the characterization data are stored in the characterization database. For stand-alone processing, all these data may be output to a summary report if the report generation option is selected in work order.

4.7.6.5 Procedure

1. For each spectral band in the L8/9 product:
    a. Extract Regions Of Interest (ROIs) defined in the associated shape file
    b. For each ROI:
        i. Generate a georeferenced thumbnail image in ENVI format
2. Using Band 2 (blue band) ROIs:
    a. Find the latitude and longitude of the region centroids
    b. Obtain the Earth-Sun distance, d, and solar elevation, θ_{SE}, and azimuth, φ, angles for the centroids using the available IAS tools
3. For each spectral band, each ROI:
    a. Count the total number of pixels, N_{Total}
    b. Flag and calculate the number of pixels in ROI that fall outside of the band image area (filled with zeros). Calculate the percentage of filled pixels in the ROI, N_{Fill}
    c. Calculate the percentage of valid pixels in the ROI, N_{Valid}. Valid pixels are currently defined as all pixels within the ROI except of filled pixels.
4. Convert the product surface temperature band DNs to surface temperature, T

$$T = M_T \cdot Q_{cal} + A_T$$

    a. For valid pixels in each ROI:
        i. Find the surface temperature minimum, T_{min}
        ii. Find the surface temperature maximum, T_{max}
        iii. Calculate the surface temperature mean, $\bar{T}$
        iv. Calculate the surface temperature standard deviation, σ_T
        v. Calculate the coefficient of variation, C_v:

$$C_v = \frac{\bar{T}}{\sigma_T}$$

5. Convert the product surface reflectance bands' DNs to surface spectral reflectance, ρ_λ

$$\rho_\lambda = M_\rho \cdot Q_{cal} + A_\rho$$

    a. For valid pixels in each band, each ROI:

i. Find the surface reflectance minimum, ρ_{min}

ii. Find the surface reflectance maximum, ρ_{max}

iii. Calculate the surface reflectance mean, $\bar{\rho}$

iv. Calculate the surface reflectance standard deviation, σ_ρ

v. Calculate the coefficient of variation, C_v:

$$C_v = \frac{\bar{\rho}}{\sigma_\rho}$$

6. For each spectral band, each ROI:

   a. Based on the results of cloud cover assessment, as reported in the Quality Band, calculate the percentage of pixels in ROI contaminated by clouds, N_{CC}

   b. Store the statistics to the database and/or report file

Appendix A Acronyms

A/D	Analog/Digital
ACS	Attitude Control System
ACTHom	Average Cross-Track Homogeneity
ADD	Algorithm Description Document
AEC	Aerosol Extinction Coefficient
ALI	Advanced Land Imager
ALIAS	Advanced Land Image Assessment System
AM	Artifact Mask
AOT	Aerosol Optical Thickness
ASCII	American Standard Code for Information Interchange
ASTER GED	Advanced Spaceborne Thermal Emission and Reflection Radiometer Global Emissivity Dataset
ATBD	Algorithm Theoretical Basis Document
ATHom	Along-Track Homogeneity
AVHRR	Advanced Very High Resolution Radiometer
B2B	Band-to-Band
BATC	Ball Aerospace and Technologies Corporation
BPF	Bias Parameter File
BRAA	Band Registration Accuracy Assessment
BT	Brightness Temperature
BUI	Bumper Mode User Interface
Cal/Val	Calibration and Validation
CCB	Configuration Control Board
CCR	Configuration Change Request
CFMask	C version of Fmask
CM	Center of Mass
CMA	Climate Modeling Grid - Aerosol
CMG	Climate Modeling Grid
CPF	Calibration Parameter File
CPT	Comprehensive Performance Test
CR	Change Request
CRC	Cyclic Redundancy Check
CTDiff	Cross-Track Difference
CTHom	Cross-Track Homogeneity
CV	Coefficient of Variation
CVT	Calibration/Validation Team
CVTK	Calibration/Validation Toolkit
DB	Database
DCE	Data Collection Event
DEM	Digital Elevation Model
DFCB	Data Format Control Book

DMA	Defense Mapping Agency
DN	Digital Number
DOI	Department of the Interior
DOQ	Digital Orthophoto Quadrangle
DSS	Data Staging System
DT	Diffuser Type
ECEF	Earth-Centered Earth-Fixed
ECF	Earth Centered Fixed
ECI	Earth-Centered Inertial
ECR	Earth Centered Rotating
ECS	EOSDIS Core System
EDU	Engineering Design Unit
EF	Earth-Fixed
ENVI	Environment for Visualizing Images
EO-1	Earth Observing-1
EOSDIS	Earth Observing System Data and Information System
EROS	Earth Resources Observation and Science
ESF	Edge Spread Function
ESPA	EROS Science Processing Architecture
ETM+	Enhanced Thematic Mapper Plus
FFOV	Full Field Of View
FFT	Fast Fourier Transform
FOV	Field of View
FP-IT	Forward Processing for Instrument Teams
FPA	Focal Plane Array
FPE	Focal Plane Electronics
FPM	Focal Plane Module
GAST	Greenwich Apparent Sidereal Time
GCP	Ground Control Point
GCRS	Geocentric Celestial Reference System
GCTP	General Cartographic Transformation Package
GED	Global Emissivity Dataset
GEOS-5	Goddard Earth Observing System Model, Version 5
GLS	Global Land Survey
GMAO	Global Modeling and Assimilation Office
GPS	Global Positioning System
GSD	Ground Sample Distance
GSFC	Goddard Space Flight Center
GSL	GNU Scientific Library
HDF	Hierarchical Data Format
HDF5	Hierarchical Data Format Version 5
HOT	Haze Optimized Transformation
Hz	Hertz

I2I	Image-to-Image
IAS	Image Assessment System
IAU	International Astronomical Union
ICD	Interface Control Document
IDL	Interactive Data Language
IEEE	Institute of Electrical and Electronics Engineers
IERS	International Earth Rotation Service
IFOV	Instantaneous Field of View
IMU	Inertial Measurement Unit
IN	Impulse Noise
IRU	Inertial Reference Unit
IS	Ingest System
ITRF	International Terrestrial Reference Frame
ITS	Integration Time Sweep
JPL	Jet Propulsion Laboratory
K	Kelvin
km	Kilometer
L0R	Level 0 Reformatted
L0Ra	Level 0 Reformatted Archive
L0Rp	Level 0 Reformatted Product
L1	Level 1
L1GT	Level 1 Systematic Terrain (Corrected)
L1R	Level 1 Radiometric (Corrected)
L1TP	Level 1 Terrain Precision (Corrected)
L7	Landsat 7
L8	Landsat 8
L9	Landsat 9
LaSRC	Land Surface Reflectance Code
LDCM	Landsat Data Continuity Mission
LM	Labeled Mask
LMDD	Landsat Metadata Description Document
LOS	Line-of-Sight
LP DAAC	Land Processes Distributed Active Archive Center
LPGS	Landsat Product Generation System
LSB	Least Significant Bit
LSCF	Landsat 8/9 Solar Calibration Frame of Reference
LSQ	Least-Squares
LUT	Lookup Table
m	Meter
mm	Millimeter
MCE	Mirror Control Electronics
MERRA-2	Modern-Era Retrospective Analysis for Research and Applications, Version 2

MIT	Massachusetts Institute of Technology
MLH	Maximum Likelihood
MLHE	Maximum Likelihood Estimate
MODIS	Moderate Resolution Imaging Spectroradiometer
MODTRAN	MODerate Resolution Atmospheric TRANsmission
MS	Multispectral
MTF	Modulation Transfer Function
MTFC	Modulation Transfer Function Compensation
NASA	National Aeronautics and Space Administration
NBR	Navigation Base Reference
NDSI	Normalized Difference Snow Index
NDVI	Normalized Difference Vegetation Index
NDWI	Normalized Difference Water Index
NEDL	Noise Equivalent Detector Radiance
NEDT	Noise Equivalent delta Temperature
NEOS	National Earth Orientation Service
NGA	National Geospatial-Intelligence Agency
NIR	Near-Infrared
NIST	National Institute for Standards and Technology
NOVAS	Naval Definition Vector Astrometry Software
NUC	Non-Uniformity Correction
OB	Orbit Reference Frame
OBC	On-Board Calibrator
ODL	Object Description Language
OLI	Operational Land Imager
OO	Orbit Oriented
PAN	Panchromatic
PCD	Payload Correction Data
PICS	Pseudo-Invariant Calibration Site
PPS	Pulse Per Second
PSF	Point Spread Function
Q	Quantized Output
QA	Quality Assessment
QWIP	Quantum Well Infrared Photometers
RIT	Rochester Institute of Technology
RLUT	Response Linearization Look-Up Table
RMSE	Root Mean Squared Error
ROI	Region of Interest
ROIC	Read-Out Integrated Circuit
ROLO	Robotics Lunar Observatory
RPC	Rational Polynomial Coefficients
SCA	Sensor Chip Assemblies
SDS	Science Dataset

SEU	Single Event Upset
SIRU	Space Inertial Reference Unit
SNR	Signal-to-Noise Ratio
SOM	Space Oblique Mercator
SR	Surface Reflectance
SSM	Scene Select Mirror
SSoH	Stored State of Health
ST	Surface Temperature
STF	System Transfer Function
SWIR	Short Wavelength Infrared
TAI	International Atomic Time
TDB	Barycentric Dynamical Time
TDT	Terrestrial Dynamical Time
TIRS	Thermal Infrared Sensor
TM	Thematic Mapper
TO	Terrain Occlusion
TOA	Top of Atmosphere
TT	Terrestrial Time
USGS	U.S. Geological Survey
UT1	Universal Time-Corrected
UTC	Coordinated Universal Time
UTM	Universal Transverse Mercator
VNIR	Visible and Near-Infrared
VRP	Video Reference Pixel
WGRIB	Web Gridded Information in Binary Form
WGS84	World Geodetic System 1984
WLS	Weighted Least Square
WRS-2	Worldwide Reference System 2

References

Please see https://www.usgs.gov/core-science-systems/nli/landsat/landsat-acronyms or for a list of acronyms.

USGS EROS Science Processing Architecture (ESPA) Elevation Project https://github.com/USGS-EROS/espa-elevation.

USGS/EROS. LAS 6.0. Geometric Manipulation Package Overview Document.

USGS/EROS. LDCM Spacecraft to Ground Interface Control Document (70-P58230P, Rev C).

USGS/EROS. LSDS-54. Landsat 7 Image Assessment System (IAS) Geometric Algorithm Theoretical Basis Document (ATBD).

USGS/EROS. LSDS-270. Landsat 7 (L7) Enhanced Thematic Mapper Plus (ETM+) Level 0 Reformatted Archive (L0Ra) Data Format Control Book (DFCB).

USGS/EROS. LSDS-524. Landsat Metadata Definition Document (LMDD).

USGS/EROS. LSDS-809. Landsat 8 (L8) Level 1 (L1) Data Format Control Book (DFCB).

USGS/EROS. LSDS-1330. Landsat Provisional Surface Temperature Product Guide, https://www.usgs.gov/media/files/landsat-provisional-surface-temperature-product-guide.

USGS/EROS. LSDS-1368. Landsat 8 Collection 1 Land Surface Reflectance Code Product Guide, https://www.usgs.gov/media/files/landsat-8-collection-1-land-surface-reflectance-code-product-guide.

Bicknell, W.E., Digenis, C.J., Forman S.E., Lencioni D.E., “EO-1 Advanced Land Imager,” MIT Lincoln Laboratory.

Akima, Hiroshi, “A New Method of Interpolation and Smooth Curve Fitting Based on Local Procedures,” Journal of the Association for Computing Machinery, Vol. 17, no. 4, October 1970.

Alexander Berk, G.P. (2005). MODTRAN 5: a reformulated atmospheric band model with auxiliary species and practical multiple scattering options: update. Defense and Security (pp. 662-667). International Society for Optics and Photonics.

Bate, R.R., Mueller D.D., and White, J.E., “Fundamentals of Astrodynamics,” Dover Publications, 1971.

Bosilovich, M.G., R. Lucchesi, and M. Suarez, 2016: MERRA-2: File Specification. GMAO Office Note No. 9 (Version 1.1)
https://gmao.gsfc.nasa.gov/pubs/docs/Bosilovich785.pdf

Brown, R.G., “Introduction to Random Signal Analysis and Kalman Filtering,” John Wiley and Sons, New York, 1983

Cook, M.J. (2014). Atmospheric Compensation for a Landsat Land Surface Temperature Product. Rochester Institute of Technology: RIT Scholar Works.

Cook, R. Dennis, “Detection of Influential Observations in Linear Regression,” Technometrics, Volume 19, Number 1, February, 1997.

DMA TR 8350.2-A, DMA Technical Report, Supplement to Department of Defense World Geodetic System 1984 Technical Report, prepared by the Defense Mapping Agency WGS84 Development Committee, dated December 1, 1987.

DMA TR8350.2-A, “Supplement to the Department of Defense World Geodetic System of 1984 Technical Report – Part I: Methods, Techniques, and Data Used in WGS 84 Development,” National Geospatial Intelligence Agency, 1 December 1987.

Goddard Earth Observing System Model, Version 5 (GEOS-5) Forward Processing for Instrument Teams (FP-IT)
https://gmao.gsfc.nasa.gov/GMAO_products/FP-IT_products.php

Glynn C. Hulley, C.G. (2012). Quantifying uncertainties in land surface temperature and emissivity retrievals from ASTER and MODIS thermal infrared data. J. Geophys. Res.

Golub, Gene H. and Charles F. Van Loan, Matrix Computations, The Johns Hopkins University Press, Baltimore, MD, 1983.

Gonzalez, R., Wintz, P., Digital Image Processing, Addison-Wesley Publishing Company, 1987.

Haykin, Simon, “Adaptive Filtering Theory,” Prentice Hall, 1991

IAS Radiometry Algorithm 6.6a, “Outliers in a Regression,” Goddard Space Flight Center, 1996.

Kaplan, George H., United States Naval Observatory Circular No. 179, “The IAU Resolutions on Astronomical Reference Systems, Time Scales, and Earth Rotation Models - Explanation and Implementation,” U.S. Naval Observatory, Washington, D.C., October 20, 2005 aka. Circular 179

Landsat 7 System Program Coordinates System Standard, Revision B, prepared by Martin Marietta Astro Space, document number PS23007610B, dated 2 December 1994.

Laraby, K. (2017). Landsat Surface Temperature Product: Global Validation and Uncertainty Estimation. Rochester Institute of Technology: RIT Scholar Works.

Lencioni, D.E., Digenis, C.J., Bicknell, W.E., Hearn, D.R., Mendenhall, J.A., "Design and Performance of the EO-1 Advanced Land Imager," MIT Lincoln Laboratory.

Lucchesi, R., 2013: File Specification for GEOS-5 FP-IT. GMAO Office Note No. 2 (Version 1.2)
https://gmao.gsfc.nasa.gov/products/documents/GEOS_5_FPIT_File_Specification_ON02_v1_2.pdf
2 (Version 1.2)

Malakar, N.K., Hulley, G.C., Hook, S.J., Laraby, K., Cook, M., & Schott, J. (2018). An Operational Land Surface Temperature Product for Landsat Thermal Data: Methodology and Validation. IEEE Transactions on Geoscience and Remote Sensing, (99), 1-19. http://dx.doi.org/10.1109/TGRS.2018.2824828.

Modern-Era Retrospective analysis for Research and Applications, Version 2 (MERRA-2)
https://gmao.gsfc.nasa.gov/reanalysis/MERRA-2/

MODerate resolution atmospheric TRANsmission (MODTRAN)
http://modtran.spectral.com/

NASA Land Processes Distributed Active Archive Center (LP DAAC)
https://lpdaac.usgs.gov/

Park, S.K., R.A. Schowengerdt, Image Reconstruction by Parametric Cubic Convolution, Computer Vision, Graphics and Image Processing, v.23, no.3, September 1983.

Press W., Teukolsky S., Vetterling W., and Flannery B., Numerical Recipes in C, 2nd edition, Cambridge University Press, 1992.

Rao, C.R., "Linear Statistical Inference and Its Applications," John Wiley and Sons, Inc., 1973.

Rosborough, G.W., D.G. Baldwin and W.J. Emery, "Precise AVHRR Image Navigation," IEEE Transaction on Geoscience and Remote Sensing, Vol. 32, No. 3., May 1994.

Sahin, M., P.A. Cross, and P.C. Sellers, "Variance Component Estimation Applied to Satellite Laser Ranging," Bulletin Geodesique, 66: 284-295, 1992.

Snyder, John P., Map Projections - A Working Manual, United States Geological Survey Professional Paper 1395, U.S. Government Printing Office, Washington, 1987.

Snyder, John P., Space Oblique Mercator Projection Mathematical Development, United States Geological Survey Bulletin 1518, U. S. Government Printing Office, Washington, 1981.

Theoretical Basis of the Science Data Processing Toolkit Geolocation Package for the ECS Project, prepared by the EOSDIS Core System Project, document number 445-TP-002-002, dated May 1995.

Web Gridded Information in Binary Form (WGRIB)
http://www.cpc.ncep.noaa.gov/products/wesley/wgrib.html

Wolberg, G., Digital Image Warping, IEEE Computer Science Press, 1990.

Yuan, Dah-ning, “Optimum Determination of the Earth’s Gravitational Field from Satellite Tracking and Surface Measurements,” Dissertation, The University of Texas at Austin, 1991.