Tampere University

Mehmet Aydin

RAW BAYER DOMAIN IMAGE ALIGNMENT

Master of Science Thesis
Faculty of Information Technology and Communication Sciences
Thesis Examiners: Prof. Moncef Gabbouj
Dr. Uygar Tuna
November 2022

ABSTRACT

Mehmet Aydin: Raw Bayer Domain Image Alignment
Master of Science Thesis
Tampere University
Signal Processing and Machine Learning
November 2022

Every day millions of photographs are captured with handheld mobile devices. Depending on the capturing conditions, multiple images may be captured in order to obtain more information on the scene in question. These multiple images are captured sequentially and called raw images. As a result, depending on the scene dynamics and due to the camera shake (while shooting), raw images will have pixel shifts. Because of that reason, during raw image capturing, images may be slightly misaligned, resulting in ghosting artifacts in the fused image. Raw images need to be aligned in order to achieve an effective fusion. Optical flow plays an important role in image alignment as it can estimate any object motion in the scene while also capturing global camera motion. It is a flow vector that determines how much each pixel of the reference frame is shifted in the moving frame. Then, using that flow vector, the moving frame can be warped to the reference frame to reduce misalignment between the reference and moving frames. The purpose of this thesis is to evaluate the performance of optical flow algorithms to objectively analyze optical flow accuracy and to assess the amount of ghosting artifacts in the fused image in raw Bayer domain. Therefore, the input data is expected to be close to a raw image. However, publicly available optical flow datasets are far from this requirement. For this reason, a new in-house raw image containing dataset has been created.

In this thesis, both conventional and deep learning-based optical flow methods were explored. The performance of the optical flow methods was evaluated on both the publicly available datasets and the in-house dataset. Based on the results obtained, both deep learning-based and conventional methods perform well when the motion between consecutive images is relatively small. However, with the increasing amount of motion, the motion estimation is more error-prone and does not perform as desired. Furthermore, the performance of optical flow algorithms was tested on different levels of noise. It has been observed that noise has a negative effect on optical flow performance. Moreover, the findings indicated that optical flow algorithms achieved remarkable overall quality improvement after applying Black Level Correction (BLC), Lens Shading Correction (LSC), White Balance (WB), and Opto-Electronic Transfer Function (OETF) to images in raw Bayer domain. Also, deep learning-based algorithms operate on RGB images. However, raw Bayer images have four channels (RGGB). Therefore, two different demosaicing algorithms were tried to reconstruct RGB images from Bayer patterns, the one green of two green channels skipping mode and bilinear interpolation. The results have shown that the alignment of the RGB images constructed by bilinear interpolation produced higher sub-pixel accuracy than the skipping mode.

Keywords: raw image, raw Bayer domain, image alignment, optical flow, deep learning, ghosting artifacts, image warping, in-House dataset, endpoint error

PREFACE

This thesis was done for Xiaomi Finland Oy. I would like to thank my supervisor Moncef Gabbouj for guiding me in my thesis work. I would like to thank my esteemed thesis advisor Dr. Uygar Tuna, who did not spare me his knowledge, experience and support during my master's thesis.

I would like to thank to director of Xiaomi Finland Oy, Dr. Jarno Nikkanen, for giving me an opportunity to work and write my thesis in parallel. I would like to thank Dr. Vladislav Uzunov for his advice and assistance during my thesis work.

I owe a debt of gratitude to my friends, Osman, Yekta, Hasem, Halit and Ismail, who stood by me on this journey and gave me support.

I would like to thank my parents, who brought me up and brought me to where I am today, I know they are always with me even though they are far away, and I cannot imagine a life without them.

Tampere, 31st November 2022

Mehmet Aydin

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

ADC	Analog to Digital Converter
AE	Angular Error
AEPE	Average Endpoint Error
AG	Analog Gain
AI	Artificial Intelligence
BLC	Black Level Correction
CC	Correlation Coefficient
CFAI	Color Filter Array Interpolation
CMOS	Complementary Metal-Oxide Semiconductor
CNN	Convolutional Neural Network
DIS	Dense Inverse Search
DSLR	Digital Single-Lens Reflex
EPE	Endpoint Error
FFN	Feed-Forward Network
FLANN	Fast Library for Approximate Nearest Neighbors
GC	Gamma Correction
GMA	Global Motion Aggregation
ILK	Iterative Lucas-Kanade
ISP	Image Signal Processor
IWS	Iterative Warping Scheme
JPEG	Joint Photographic Experts Group
LID	Local Invariant Descriptor
LIDAR	Light Detection and Ranging
LK	Lucas-Kanade
LSC	Lens Shading Correction
MHA	Multi Head Attention
NLP	Natural Language Processing

OETF	Opto-Electronic Transfer Function
PNG	Portable Network Graphics
PSNR	Peak Signal-to-Noise Ratio
RAFT	Recurrent All Pairs Field Transforms
RGB	Red-Green-Blue
RNN	Recurrent Neural Network
SAM	Spectral Angle Mapper
SIFT	Scale Invariant Feature Transform
SOTA	State-of-the-Art
SPD	Spectral Power Distribution
TBN	Transformer Based Network
TV	Total Variation
UQI	Universal Quality Image Index
WB	White Balance

1. INTRODUCTION

Mobile phone imaging has grown significantly in recent years. This has made the camera the most important part of the phone and lots of research has been done to improve the image quality.

Recently, the advancements in smartphone cameras have been concentrated on improving the low light performance to approach the level of professional photography in challenging lighting conditions. The fundamental issue in low light photography is the low signal-to-noise ratio. While this can be addressed using more advanced hardware (i.e., imaging sensors), smartphone cameras pose size limitations. Therefore, there have been efforts to increase the signal-to-noise ratio computationally. One of the ways to do so is to capture raw Bayer image sequences in quick succession and fuse them computationally in the raw Bayer domain.

The typical method of reconstructing the final image from raw Bayer images is to perform a set of image processing algorithms in Image Signal Processor (ISP). A typical set of image processing algorithms are Black Level Correction (BLC), Lens Shading Correction (LSC), White Balancing (WB), demosaicing, color correction, gamma correction (mainly called Opto-Electronic Transfer Function (OETF)), and tone mapping, as shown in Fig. 1.1.

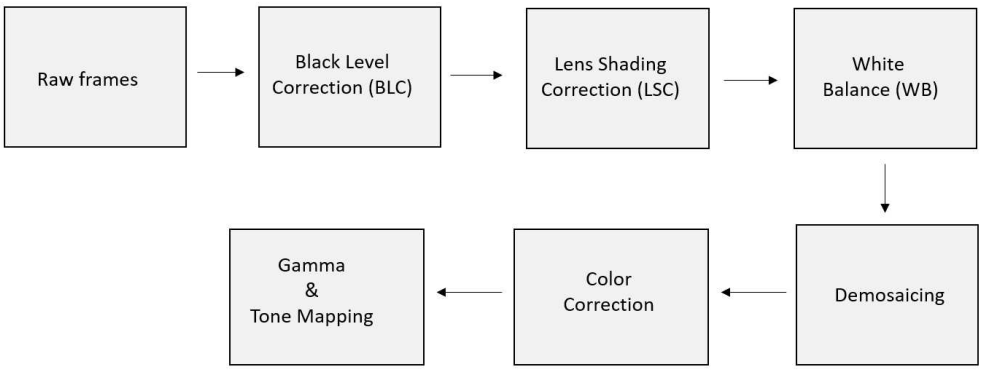
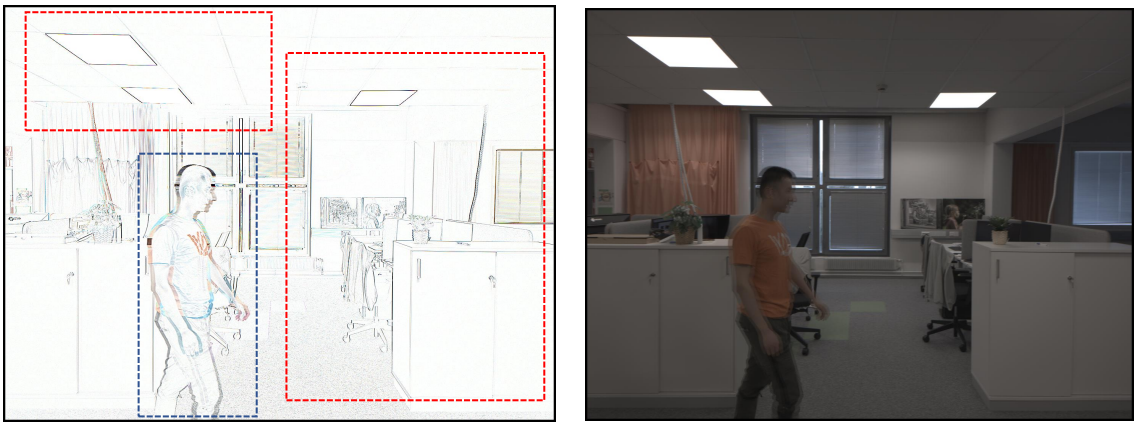


Figure 1.1. *A simplified example image signal processor pipeline. The order of the blocks may vary depending on the design of the ISP.*

As mobile phones are always handheld, raw images will have pixel shifts due to hand-shaking and possible object(s) movement in the scene, which causes the images to be

slightly misaligned. An example of misalignment between two consecutive processed raw images is illustrated in Fig. 1.2a. The dominant motion in the regions indicated by the red rectangles occurs due to the global camera motion (also known as camera shake or hand tremor), while the dominant motion in the region indicated by the blue rectangle occurs due to the object motion. That makes it difficult to fuse information from multiple images effectively. Ghosting is the most common artifact encountered when fusing images that are not perfectly aligned. An example of ghosting artifact is depicted in Fig. 1.2b.



(a) ***(b)***

Figure 1.2. *(a) Misalignment between two consecutive processed raw images due to global camera motion (the regions indicated by the red rectangles) and object movement (the region indicated by the blue rectangle) in the scene. (b) Ghosting artifacts due to the pixel shift.*

Raw images need to be aligned to avoid ghosting artifacts in the fused image as shown in Fig. 1.3. There have been many methods to align images. They can be divided into two categories: feature-based and pixel-based image alignment.

Feature-based image alignment relies on finding corresponding features in the reference and target frames to create a global transformation matrix, called homography, and then use this matrix to warp the target frame to the reference frame to be aligned. On the other hand, pixel-based image alignment relies on computing optical flow to determine how much each reference frame pixel is shifted in the target frame. The result of optical flow is a vector and it is used to warp the target frame to the reference frame to be aligned.

Optical flow plays an important role in image alignment since it estimates the motion between consecutive frames for each pixel globally and locally. The concept of optical flow was first presented by Gibson [1]. The first novel framework of optical flow was introduced by Horn and Schunck [2], which is based on variational theory [3] to compute optical flow. Lucas-Kanade proposed a method for optical flow computation aiming to minimize the cost function based on pixel difference [4]. Different traditional methods were developed over time [5, 6, 7, 8]. On the other hand, optical flow estimation has

recently received huge attention in deep learning. Convolutional Neural Network (CNN) has been used firstly by Dosovitskiy et al. for the optical flow estimation [9]. Sun et al. proposed another CNN-based optical flow estimation method, PWC-Net, which adopts a warping layer to compute optical flow from coarser to fine level [10]. Bhat et al. [11] used PWC-Net to estimate sub-pixel shifts between consecutive raw burst images to align them to a reference frame. Jiang et al. [12] used the transformer architecture [13] for the first time for the optical flow task to improve the motion estimation in occluded region in the scene. Huang et al. [14] proposed a transformer based architecture to estimate optical flow and achieved state-of-the-art (SOTA) performance on the Sintel dataset [15]

In this thesis, different optical flow algorithms were evaluated, deep learning-based and traditional methods, to estimate pixel shifts between raw images. Furthermore, the potential applicability of optical flow algorithms in raw Bayer domain was investigated. The image alignment and fusion pipeline used in this thesis is illustrated in Fig. 1.3 and follows the following steps in order.

1. The input of the image alignment and fusion pipeline is raw Bayer images with four channels (RGGB). However, deep learning-based optical flow algorithms operate on RGB images. Therefore, two different demosaicing algorithms were tried to reconstruct RGB images from Bayer patterns, the one of the two green channels skipping mode, where the one of the two green channels is discarded and bilinear interpolation, where the missing color channels are interpolated [16].
2. Select the reference image among the raw images. For convenience, the middle image is chosen as reference. Because it is equidistant from the other images, may provide a more accurate result.
3. Select a target raw image to be aligned.
4. Compute the optical flow to determine how much each pixel in the reference image is displaced in the target image so that the corresponding image content can be perfectly aligned.
5. Warp the target image to the reference image using optical flow output to align them.
6. Fuse aligned target image and reference image.
7. repeat 2nd, 3rd, 4th, and 5th steps for every target images to finally get a single fused raw image to be further processed in ISP.

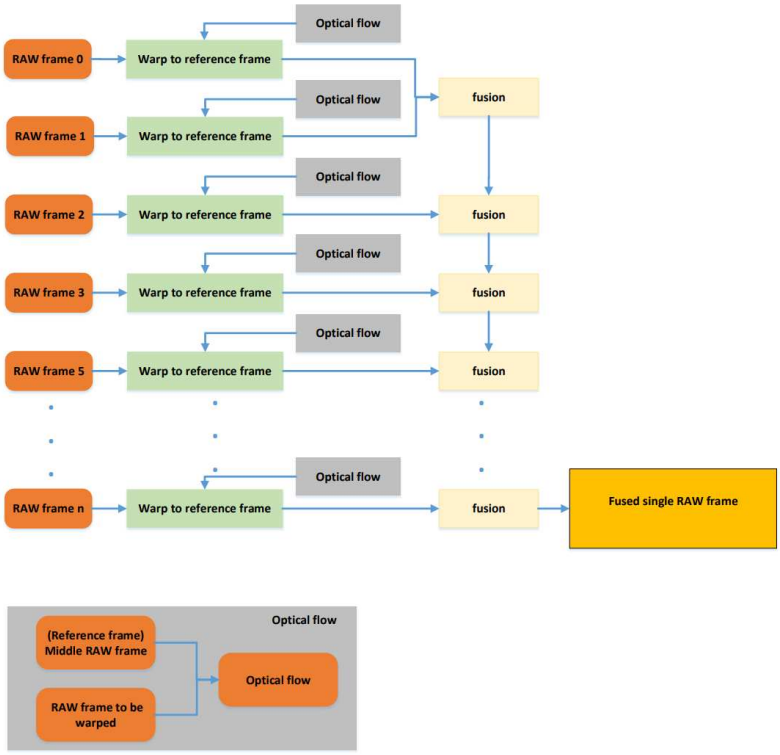


Figure 1.3. *Raw bayer domain image alignment and fusion pipeline.*

This thesis organized as follows. The raw Bayer domain image processing is presented in Chapter 2. Chapter 3 provides basic information about deep learning algorithms used in optical flow estimation tasks. Chapter 4 covers the theoretical basics of motion estimation, which the thesis is mostly built on. Chapter 5 explains image warping and fusion. Chapter 6 presents the datasets used for optical flow estimation tasks. Chapter 7 introduces the evaluation metrics used to assess the quality of the optical flow and image fusion. The results are provided and discussed in Chapter 8. Finally, Chapter 9 concludes the main findings and possible further work.

2. RAW BAYER DOMAIN IMAGE PROCESSING

This chapter introduces image processing in raw Bayer domain. Image Signal Processor (ISP) pipeline shown in Fig. 2.1b is presented in Section 2.1 and the blocks of the image alignment and fusion module illustrated in Fig. 2.1b is discussed in Section 2.2.

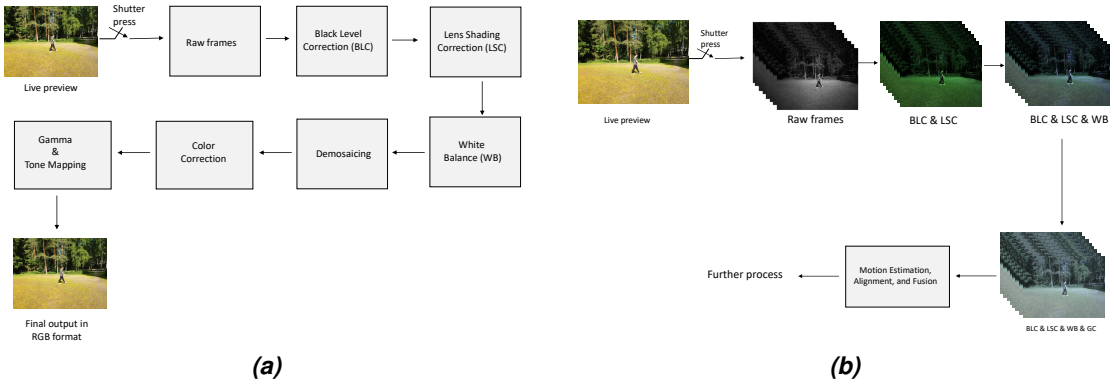


Figure 2.1. *(a) A simplified example ISP pipeline. (b) Image alignment and fusion module in ISP pipeline.*

2.1 Image Signal Processor pipeline

Image Signal Processor (ISP) is a processor integrated into the camera to take care of the specific tasks of a camera, such as white balancing, multi-frame alignment, and noise reduction.

Raw Frames

Raw image is recorded in a raw-image format in which no image information is lost since it is saved as it comes out of the sensor without compression. An example of raw Bayer image is illustrated in Fig. 2.2 that shows the Bayer pattern clearly. On the other hand, Fig. 2.3 shows how the image will look when no color processing is applied in addition to Color Filter Array Interpolation (CFAI). As seen, the image is greenish because of the combination of illumination Spectral Power Distribution (SPD) and the sensor spectral response [17].

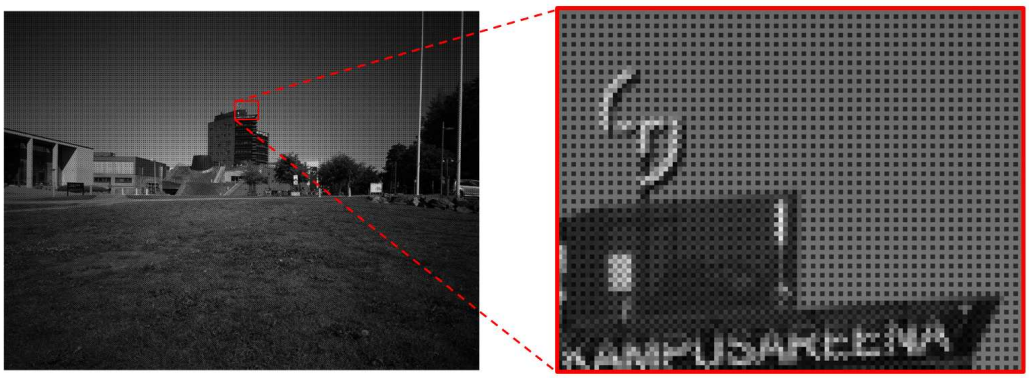


Figure 2.2. *Illustration of a raw Bayer image.*

Figure 2.3. *An example image with no color processing.*

Black Level Correction

Raw image can have non-zero black level, and this black level needs to be removed if linear signal is needed in processing, and hence it is usually removed early in ISP like shown in Fig. 1.1 and 2.1. Fig. 2.4 depicts the effect of BLC.

(a) Image as in Fig. 2.3 but Opto-Electronic Transfer Function (OETF) applied.

(b) Image as in Fig. 2.3 but BLC and OETF applied

Figure 2.4. *Impact of BLC*

Lens Shading Correction

Lens shading correction is to solve the shadow around the lens. Shading can be subdivided into luma shading and color shading. Due to the optical characteristics of the lens, the edge area of the sensor image area receives less light than the center, resulting in inconsistencies in the brightness of the center and all four corners called luma shading. Color shading (sometimes called color uniformity) is usually defined as a color shift from the center of the image towards the corners. Color shading characteristically occurs in images taken with a smaller sensor camera. A flat-field image is used to measure the lens shading. Correction information is estimated from these flat-field images to remove the shade from the lens. Fig. 2.5 shows how Fig. 2.4b looks like after lens shading correction. It can be seen that shading at the edge of Fig. 2.4b is removed after LSC applied.

Figure 2.5. *Image as in Fig. 2.3 but BLC, LSC, and OETF applied.*

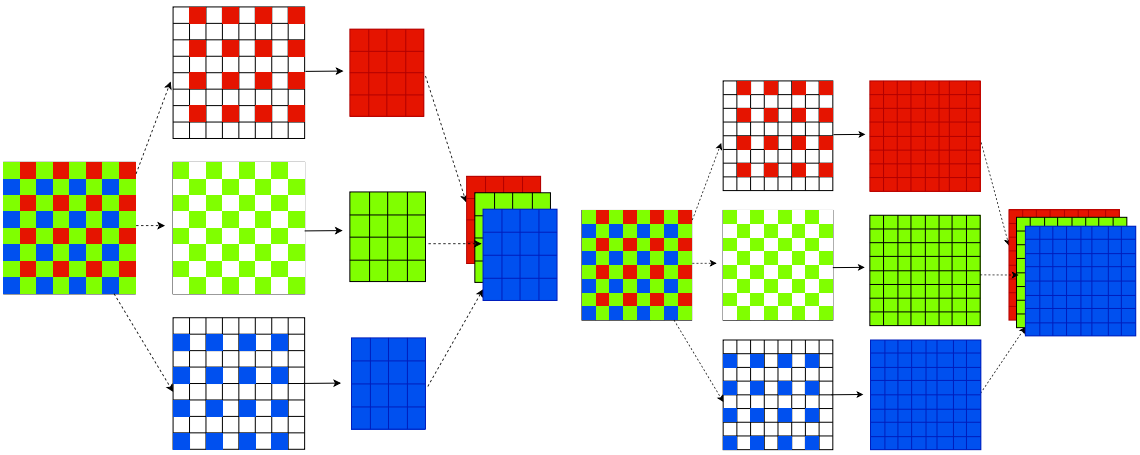
White Balance

The white balance is considered one of the other important step in ISP pipeline to obtain the real colors of a scene. Light sources produce different SPD that lead to different camera responses to the same scene contents. Although the human eye can be automatically adjusted to various light sources and color temperatures to detect the actual color, the camera sensor alone cannot automatically detect the actual color. Therefore, the image needs to be processed to reproduce the exact color. For this reason, white balance is essential to adjust the tones of colors according to the current light conditions. Fig. 2.6 illustrates the impact of WB.

Figure 2.6. *Image as in Fig. 2.3 but BLC, LSC, WB, and OETF applied.*

Demosaicing

Each pixel of the fused raw image contains a single color component. Demosaicing is applied to reconstruct the other two color components. In this thesis, two different demosaicing algorithms were tried. The first one is the one of the two green channels skipping mode, where the one of the two green channels is discarded. This causes the image resolution to drop in half. In other words, 2x2 pixels Bayer quad is considered 1x1 pixel in RGB domain as shown in Fig. 2.7a. In addition, it does not correct the 1 pixel shift in the spatial location of each color component. In the 2x2 Bayer quad, the R, G, G, and B pixels correspond to different spatial locations on the camera sensor. If this 2x2 pixel Bayer quad is treated as RGB pixel by dropping one of the green channels, then this 1 pixel phase difference remains on top of halving the resolution. The second one is linear interpolation, where the missing color components are interpolated for each pixel location as shown in Fig. 2.7b. The resulting RGB image has the same resolution as the raw bayer image.



(a) Demosaicing with one of the two green channels skipping mode

(b) Demosaicing with bilinear interpolation

Figure 2.7. *Illustration of demosaicing for reconstructing the other two color components*

Color Correction

Color Correction is an important part of ISP. It converts the RGB data from sensor RGB color space into device independent target sRGB color space by multiplying each pixel in the sensor RGB with a 3x3 matrix. Fig. 2.3 with BLC, LSC, WB, color correction, and OETF is illustrated in Fig. 2.8.

Figure 2.8. *image as in Fig. 2.3 but BLC, LSC, WB, color correction, and OETF applied.*

Gamma Correction and Tone Mapping

Gamma correction is mainly called Opto-Electronic Transfer Function (OETF), which is the transfer function from camera sensor color space to the device independent target color space. Display processing has its own transfer functions to map the data from

device independent color space to the displays own color space, so that the color look correct. On the other hand, tone mapping is the process to increase the dynamic range and also map it into the range of observing media.

2.2 Image Alignment and Fusion Pipeline in ISP

In ISP pipeline, raw images are aligned and fused after applying BLC. It is observed that most of the optical flow algorithms can not achieve the desired performance when motion is estimated after applying BLC. However, the findings have showed that optical flow algorithms achieve a remarkable overall quality improvement after applying LSC, WB, and OETF to images, in addition to BLC. After all, the optical flow is computed as shown in Fig. 2.1b.

3. DEEP LEARNING

Deep learning is a set of methodologies that allows predicting the outputs upon giving complex input data. The input data usually consists of images, text, or audio. Deep learning is inspired from the way the human brain process information. One purpose of deep learning is to simulate human-like decision making. Neurons in the brain transmit signals to perform actions. Similarly, artificial neurons are connected to a neural network to perform clustering, classification, or regression tasks. Deep learning is used in healthcare, advertising, the autonomous industry, translation, and chatbots [18, 19, 20, 21, 22]. The main advantage of deep learning is that the architecture is flexible enough to easily adapt to new problems, while the main disadvantages of deep learning are that it is quite expensive to build a deep learning architecture to train due to the data complexity, and massive amount of data is required to deliver the best results.

A simple architecture of the neural network is shown in Fig. 3.1. A simple network includes three layers: input layer, hidden layers, and output (target) layer. Layers are connected via nodes. Data is first fed into the neural network via the input layer. The node multiplies the inputs with appropriate weights. Input-weighted products are then summed, and bias added. Finally, the sum is passed through the activation function to decide whether its input to the network is important or not, which affects the final output.

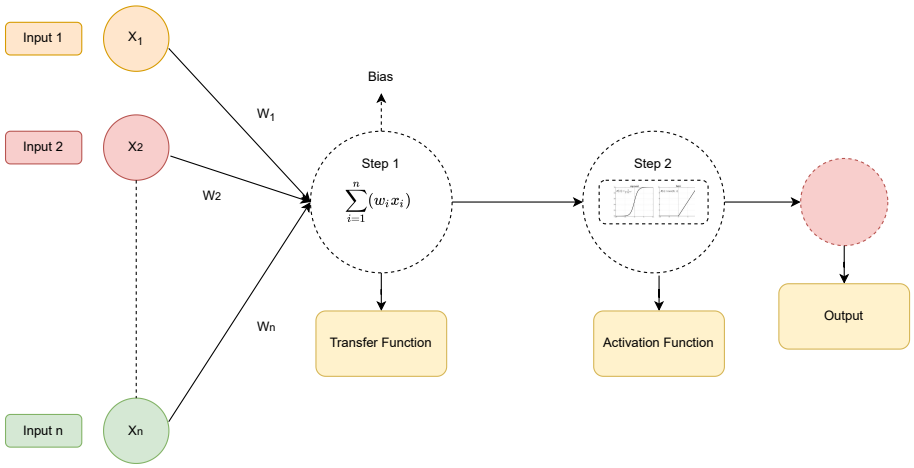


Figure 3.1. *Simple architecture of neural network.*

This chapter provides basic information about deep learning algorithms, which are used in optical flow estimation tasks. Section 3.1 presents the most commonly used deep learning algorithm Convolutional Neural Network (CNN) and its architectures. Then, Recurrent Neural Network (RNN) is described in Section 3.2. Finally, Section 3.3 explains Transformer-based network.

3.1 Convolutional Neural Network

Convolutional Neural Network (CNN) or ConvNets is a deep learning model that is often applied to analyze visual imagery that can take images as input and assign weights and biases to different patterns in the input to distinguish one from the other. CNN can be applied to Natural Language Processing (NLP), audio, and image processing [23, 24, 25]. There are three types of layers that CNN architectures usually built on: convolutional layer, pooling layer, and fully connected layer as shown in Fig. 3.2.

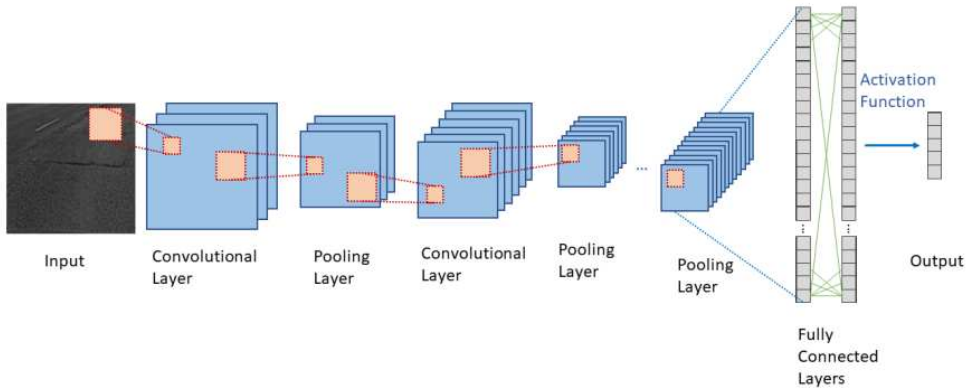


Figure 3.2. *Typical CNN architecture obtained from [26].*

Convolutional layers are used in CNN to perform feature extraction from input, and it is where the majority of the computations occur. A convolution layer includes filter that perform the convolution operation.

The pooling layer takes care of reducing the size of the feature maps. By using pooling, the best features are selected and transmitted. Pooling layers are divided into two parts depending on their purpose, which are maximum pooling and average pooling. Maximum pooling selects the highest signal in the feature map, and it is generally used in image classification problem statements, while average pooling averages all the signals in the feature map and is commonly used in image reconstruction or segmentation.

The fully connected layer is the last layer of CNN, where the classification and identification tasks are performed based on extracted features through the convolutional neural network.

Different types of CNN architectures were developed in recent years to solve real-world problems. The first CNN architecture LeNet[27] was introduced by Yann LeCun in 1998 to

recognize handwritten digits in images. Alexnet[28] architecture was developed in 2012, which significantly increased the classification accuracy of the ImageNet [29] and pioneered faster training of CNNs. VGGNet [30] architecture was presented by Visual Geometry Group Lab (VGG) of Oxford University in 2014. The high kernel sizes used in the AlexNet architecture have been reduced, and fixed its kernel dimensions which are not fixed in AlexNet. Autoencoders [31, p. 499-501] architecture consists of two parts: encoder and decoder. The dimensionality of the input data is reduced by the encoder, while the decoder is responsible for reconstructing the input data to the original size. U-Net [32] architecture is used in many pixel-to-pixel mapping, mimicking the processing using image-scales. The network has an encoder and a decoder part.

The comparison of the CNN architectures by the number of parameters, where M refers to million, number of layers, and architecture description is given in the Table 3.1

Table 3.1. *The comparison of CNN architectures. Conv refers to convolutional layer, and fc refers to the fully connected layer*

Architecture	Number of parameters (M)	layers	Architecture description
LeNet	0.06	7	5 conv + 2 fc layers
AlexNet	61	8	5 conv + 3 fc layers
VGGNet16	138	16	13 conv + 3 fc layers
VGGNet19	144	19	16 conv + 3 fc layers

3.2 Recurrent Neural Network

Recurrent Neural Network (RNN) is a type of artificial neural network that specializes in processing sequential data or time series [33]. The most important feature that distinguishes RNNs from other deep learning techniques is that RNNs can remember all information about what has been computed up to time step t as shown in Fig. 3.3. In other words, RNNs always have the information of the time step or the previous times in the time transition. RNNs are widely used in image processing and language processing [34, 35]. The main advantage of RNN, as mentioned, is that they can remember each information over time, while one of disadvantage is the problems of vanishing and exploding gradient [36].

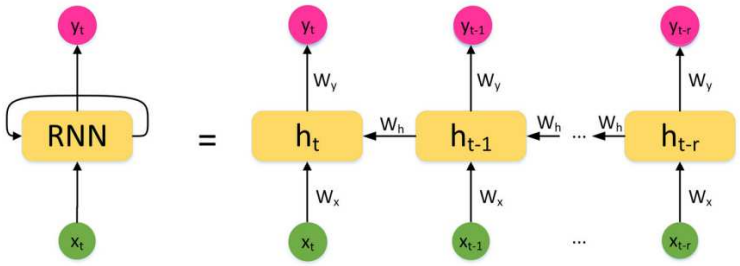


Figure 3.3. *Unfolded RNN architecture at time t obtained from [37]. The horizontal arrows connect the previous hidden layer to the current hidden layer.*

3.3 Transformer-Based Network

Transformer-Based Network (TBN) is a family of deep learning models that have been using the basic building block of NLP architectures in recent years [13]. It is a simple network architecture based on attention mechanisms. It aims to selectively focus on particular parts of the inputs and thus learn faster and more effectively.

The encoder architecture is shown on the left of Fig. 3.4, and the decoder architecture is on the right. They consist of modules that can be interconnected multiple times, as explained by Nx in Fig. 3.4. The modules mainly consist of Multi-Head Attention (MHA) and Position-wise Feed-Forward Network (FFN) layers. TBNs are used in machine translation tasks, sequence analysis and optical flow estimation [12, 14, 38, 39]. One of the advantage of the TBN is that it relies on an attention mechanism that enables AI models to selectively focus on certain parts of their input, thus learning more effectively. One of the disadvantage of the TBN is that it is computationally expensive due to the need for performing attention operations which have second-order time and space complexity with respect to the context size [40].

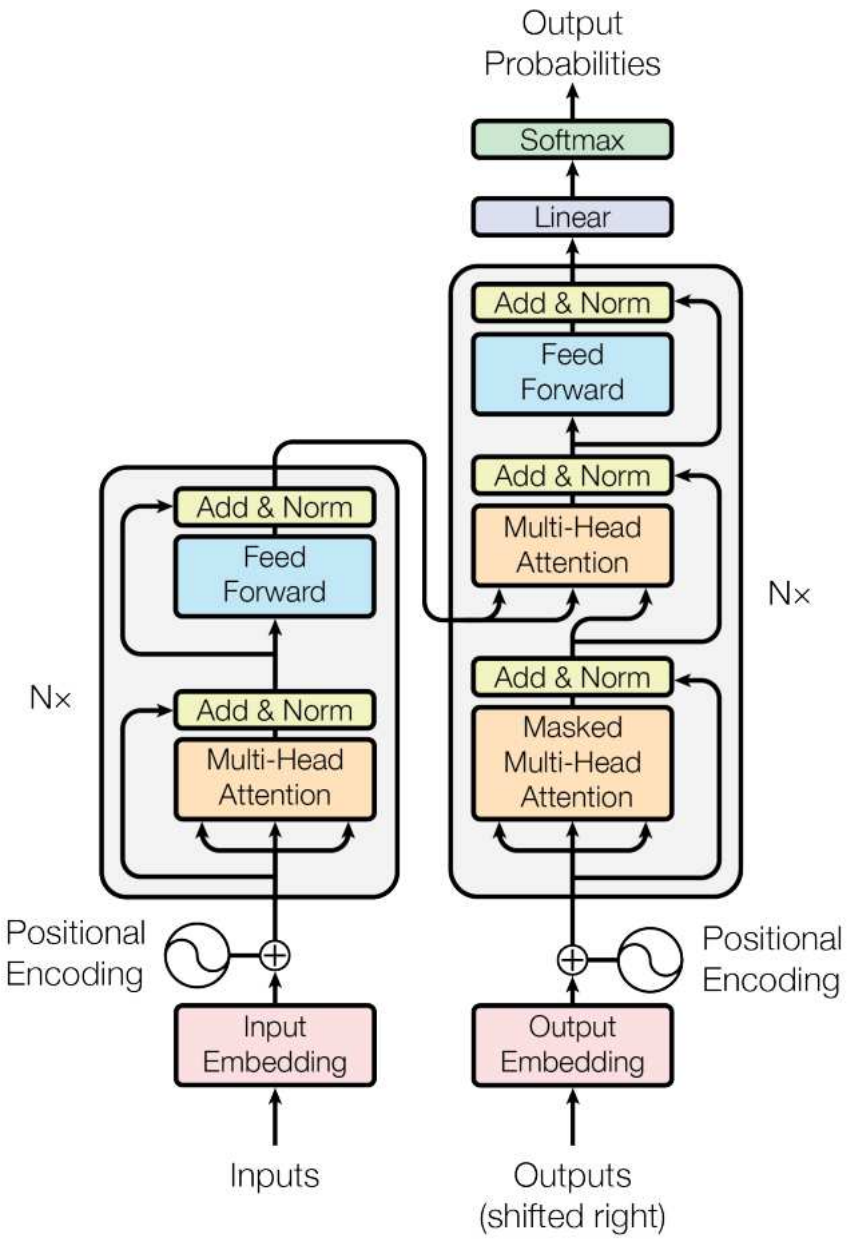


Figure 3.4. *The architecture of Transformer obtained from [13].*

4. MOTION ESTIMATION

This chapter describes the theoretical basics of motion estimation on which the thesis is mostly built. Section 4.1 introduces projective transformation. Then, Section 4.2 and Section 4.3 describe motion estimation methods: feature-based and pixel-based motion estimation, respectively.

4.1 Projective Transformation

Projective transformation is sometimes called perspective transformation or homography. This transformation works in homogeneous coordinates and preserves straight lines. A 3x3 arbitrary matrix can express this transformation in homogeneous coordinates p and p',

$$\mathrm{p}' = \mathrm{H}\mathrm{p} \tag{4.1}$$

where H is 3x3 arbitrary matrix.

Projective transformation supports the other 2D (planar) transformations: translation, Euclidean, similarity, and affine. Graphical representation of 2D transformations is shown in Fig. 4.1. It shows how the current image looks like after applying the transformation compared to the reference image.

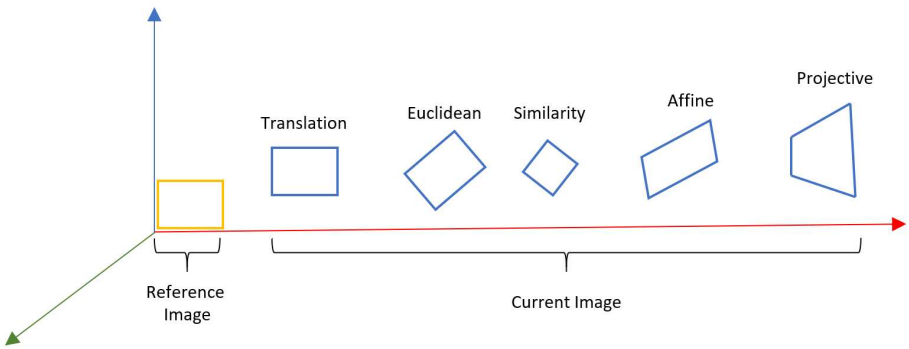


Figure 4.1. *Graphical representation of 2D (planar) transformations.*

4.2 Feature-Based Motion Estimation

Feature-based motion estimation relies on finding corresponding features in two different images. It aims to match these features to build a global correspondence. Then, the geometric transformation between these two images is estimated. Steps followed in this thesis for feature-based motion estimation is shown in Fig. 4.2.

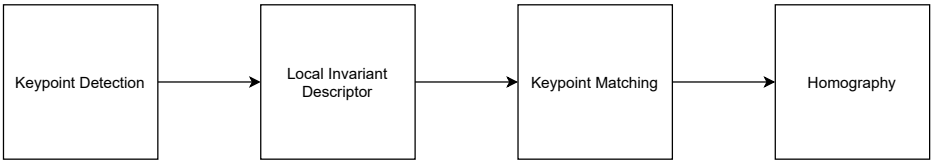


Figure 4.2. *Steps followed in this thesis for feature-based motion estimation.*

Section 4.2.1 presents keypoint detection methods. Then, Local Invariant Descriptor (LID) is explained in Section 4.2.2. Next, Section 4.2.3 explains keypoint matching. Finally, Section 4.2.4 describes homography and motion model estimation.

4.2.1 Keypoint Detection

Keypoint detectors are used to detect the distinctive features such as corners, edges, and line intersections. Harris corner detection and Scale Invariant Feature Transform (SIFT) are methods used for keypoint detection.

4.2.1.a Harris Corner Detection

Harris corner detection [41] uses a second-moment matrix to detect corners. The second moment matrix is defined as $\mathrm{M_I} = \mathrm{M_I}(\mathrm{p})$ on a selected point (p) on image plane.
The second moment matrix is

$$\mathbf{M}_\mathrm{I} = \nabla \mathrm{I} \nabla \mathrm{I}^\mathrm{T} = \begin{bmatrix} \mathrm{I_x} \\ \mathrm{I_y} \end{bmatrix} \begin{bmatrix} \mathrm{I_x} & \mathrm{I_y} \end{bmatrix} = \begin{bmatrix} \mathrm{I_x^2} & \mathrm{I_x I_y} \\ \mathrm{I_x I_y} & \mathrm{I_y^2} \end{bmatrix} \tag{4.2}$$

where $\nabla \mathrm{I}$ is the image gradient including the derivative of the image with respect x and y directions ($\mathrm{I_x}$ and $\mathrm{I_y}$).

There are three conditions of the eigenvalues of the second-moment matrix to define a point as flat, corner, and edge, depending on the gradient values in the region as shown in Fig. 4.3. If both eigenvalues λ_1 and λ_2 of the matrix M are large, then it can be said that point p is a corner. If λ_1 is large and λ_2 is close to zero, then point p lies on edge. If both eigenvalues are zero or close to zero, then point p is located in a flat area. Fig. 4.6a shows an example of Harris corner detector.

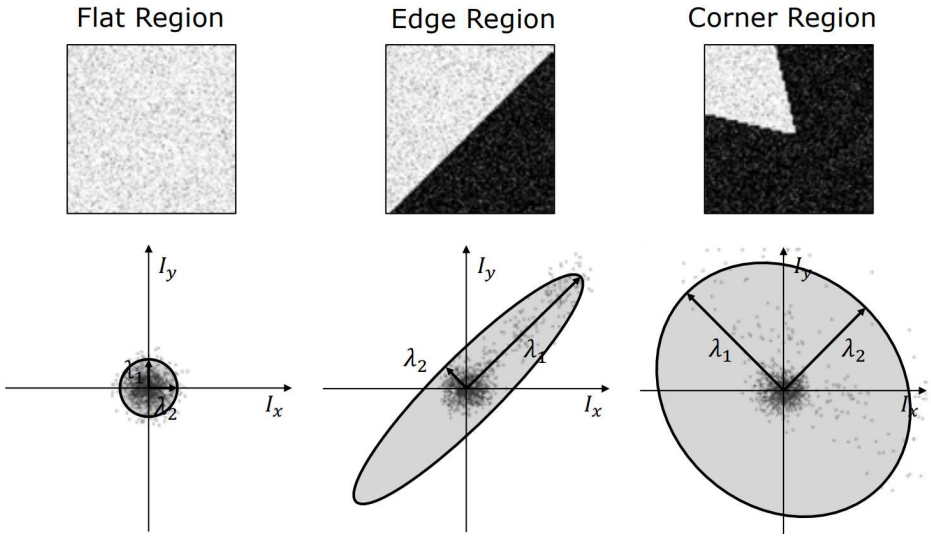


Figure 4.3. *Distribution of* $\mathrm{I_x}$ *and* $\mathrm{I_y}$ *in flat, edge, and corner regions obtained from [42].*

4.2.1.b Scale Invariant Feature Transform for Keypoint Detection

Scale Invariant Feature Transform (SIFT) is another method to detect the keypoints [43]. The biggest advantage of SIFT on Harris detector is that it is scale and rotation invariant detector, which means that SIFT can perform well if the image is scaled differently. However, Harris detector can not perform well. Graphical representation of SIFT keypoint detection algorithm with six Gaussian image layers is illustrated in Fig. 4.4. The first thing here is to create a stack of images, where the source image is blurred at each scale. Then, keypoints of SIFT can be found by using Difference-of-Gaussian (DoG) functions given in equation 4.3.

$$\mathrm{D}(\mathbf{x}, \sigma) = [\mathrm{L_{k\sigma}}(\mathbf{x}) - \mathrm{L_\sigma}(\mathbf{x})]\, \mathrm{I}(\mathbf{x}) = [\mathrm{L_{k\sigma}} - \mathrm{L_\sigma}]\, \mathrm{I} = \mathrm{I_{k\sigma}} - \mathrm{I_\sigma} \tag{4.3}$$

Where $\mathrm{L_\sigma}$ represents a 2D Gaussian kernel, $\mathrm{I_\sigma}$ a $\mathrm{L_\sigma}$ -blurred grayscale image, and k is a constant.

Keypoints are the maximums and minimums of $\mathrm{D}(\mathbf{x}, \sigma)$ across both image location and scale. The keypoints lying on edges are removed by filtering these candidate locations.

The extremum values are computed by

$$z = -\frac{\partial^2 \mathrm{D}^{-1}}{\partial \mathrm{x}^2} \frac{\partial \mathrm{D}}{\partial \mathrm{x}} \tag{4.4}$$

Where D is the difference-of-gaussian computed in equation 4.3. The value of r is computed to remove the keypoints lying on edges. If the threshold value is lower than the value of r, then that point is excluded. Fig. 4.6b shows an example of detected keypoints by SIFT where the circle size directly indicates the strength of the keypoints.

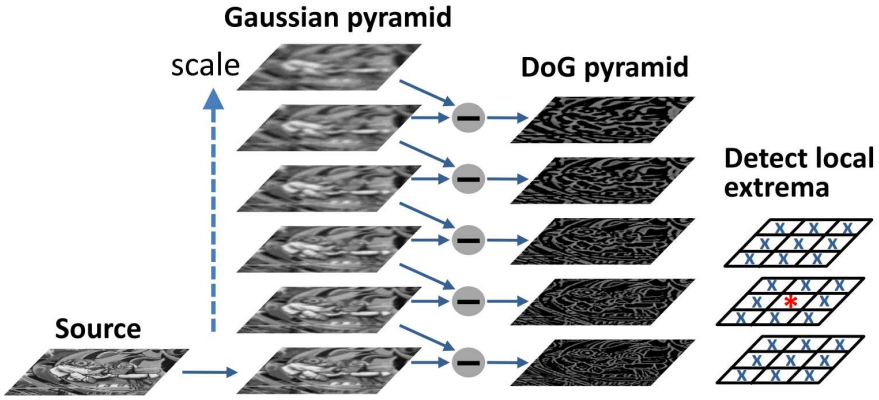


Figure 4.4. *Graphical representation of SIFT keypoint detection algorithm with six Gaussian image layers obtained from [44].*

4.2.2 Local Invariant Descriptors

Local Invariant Descriptors (LIDs) are usually used for quantifying the region surrounding each keypoint. The commonly used descriptors are SIFT, Speeded Up Robust Features (SURF), [45], and Oriented FAST and Rotated BRIEF (ORB) [46]. In this thesis, SIFT is used as a descriptor. According to [43], SIFT creates region by taking 16x16 neighborhood around the keypoint for 233 by 189 pixel image as shown on the left of the Fig. 4.5, where the arrows indicate the magnitude of the gradient and direction of each image sample point located in 16x16 block. This region is then divided into 16 blocks of 4x4 dimensions. An 8 bins orientation is generated for each sub-block as illustrated on the right of the Fig. 4.5. So in total, 128 bin values (16 sub-blocks x 8 bins per-block) are created as a vector to obtain the keypoint descriptors.

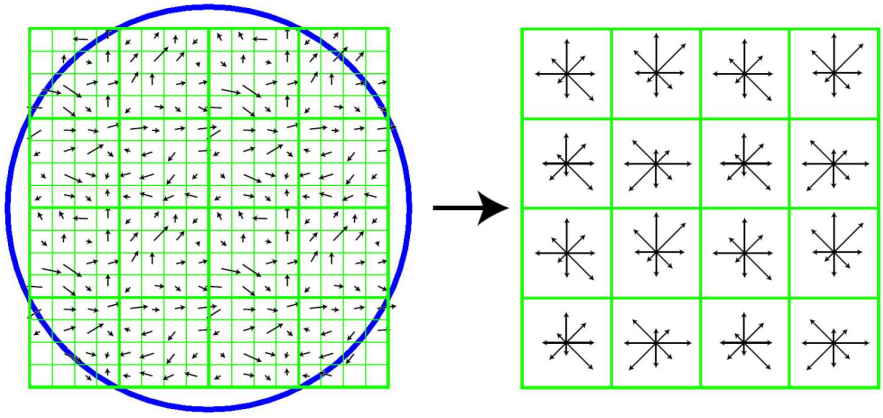


Figure 4.5. *Keypoint descriptors with SIFT obtained from [47].*

4.2.3 Keypoint Matching

Once features and descriptors are extracted from the images, the feature of an image needs to be matched to feature of a target image. In other words, the idea is to determine a set of correspondences between descriptors in two images. Some of the well known methods are brute-force matcher [48] and Fast Library for Approximate Nearest Neighbors (FLANN) [49]. In this thesis, FLANN method is used to match the same physical points between two images. It includes a set of algorithms optimized for fast nearest neighbor searches. Once the matches are found, then a set of inlier correspondence needs to be found that will produce a highly accurate keypoint matching by using a threshold to eliminate false or bad matching points. In N match pairs, sample sets with an error range less than the given threshold are considered true match pairs, and the rest of the matching pairs are considered outliers or called false match points. An example of keypoints matching of two images is shown in Fig. 4.6c

4.2.4 Homography and Motion Model Estimation

Having enough matching of keypoints, a transformation matrix is computed to estimate the global motion between two images. This transformation matrix is called Homography which relates to the transformation between two images.

As mentioned in section 4.1, homography is a 3x3 matrix as seen in equation 4.5.

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{4.5}$$

Homograpy function is defined to relate a point (x_1, y_1) in the first image and the same physical point (x'_1, y'_1) in the transformed version of the first image as seen in equation 4.6

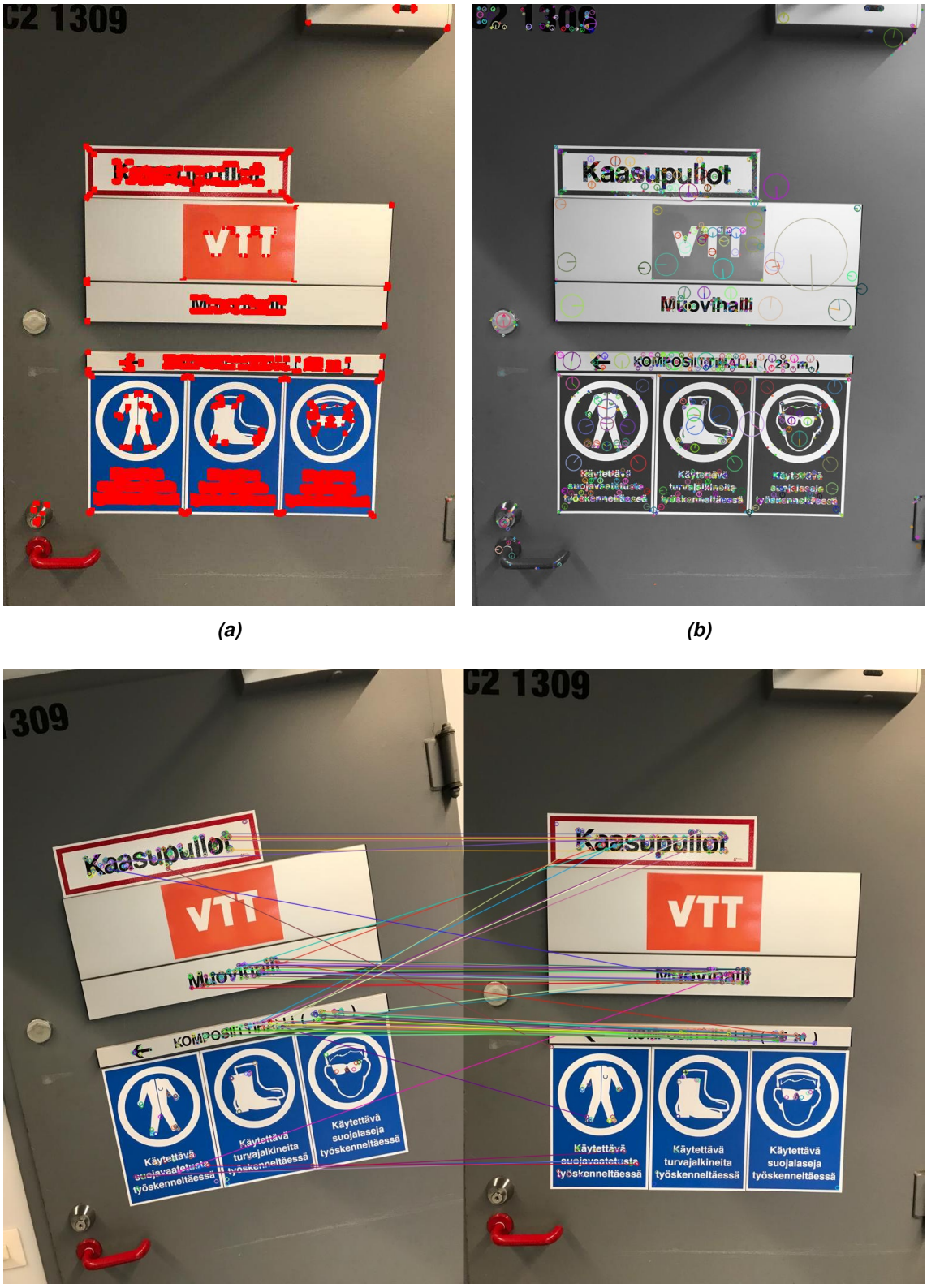
$$\begin{bmatrix} x'_1 \\ y'_1 \\ 1 \end{bmatrix} = H \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \tag{4.6}$$

As can be seen from equation 4.6, there is 9 unknown parameters that needs to be estimated. The equation 4.6 can be written as linear equation

$$\begin{bmatrix} -x_1 & -y_1 & -1 & 0 & 0 & 0 & x_1x_1' & y_1x_1' & x_1' \\ 0 & 0 & 0 & -x_1 & -y_1 & -1 & x_1y_1' & y_1y_1' & y_1' \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{4.7}$$

Homography has 8 degrees of freedom. To solve this, the minimum of pairs of matching point is 4, and the last parameter of homography (h_{33}) is considered as 1. If all rows are stacked corresponding to these 4 pairs of matching point, the homography parameters then can be computed as given in equation 4.8.

$$H = \begin{bmatrix} -x_1 & -y_1 & -1 & 0 & 0 & 0 & x_1x_1' & y_1x_1' & x_1' \\ 0 & 0 & 0 & -x_1 & -y_1 & -1 & x_1y_1' & y_1y_1' & y_1' \\ -x_2 & -y_2 & -1 & 0 & 0 & 0 & x_2x_2' & y_2x_2' & x_2' \\ 0 & 0 & 0 & -x_2 & -y_2 & -1 & x_2y_2' & y_2y_2' & y_2' \\ -x_3 & -y_3 & -1 & 0 & 0 & 0 & x_3x_3' & y_3x_3' & x_3' \\ 0 & 0 & 0 & -x_3 & -y_3 & -1 & x_3y_3' & y_3y_3' & y_3' \\ -x_4 & -y_4 & -1 & 0 & 0 & 0 & x_4x_4' & y_4x_4' & x_4' \\ 0 & 0 & 0 & -x_4 & -y_4 & -1 & x_4y_4' & y_4y_4' & y_4' \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{4.8}$$



(a) *(b)*

(c)

Figure 4.6. *(a) An example of corner detection by the Harris corner detector. Red points denote the detected corners.(b) An example of detected keypoints by SIFT where the circle size indicates the strength of the keypoint and line shows the orientation of the key point.(c) Matching of keypoints between an image and transformed version of the same image using FLANN method.*

4.3 Pixel-Based Motion Estimation

Section 4.3.1 describes and formulates the optical flow with the brightness constancy assumption. Furthermore, it defines the dense optical flow. Then, Section 4.3.2 presents the conventional and deep-learning-based methods to compute the dense optical flow in this thesis.

4.3.1 Optical Flow

When the object moves in the 3D scene, the motion in the image is called optical flow. Optical flow defines the direction and speed of motion in the image. Optical flow estimation is used in object tracking [50, 51], action recognition [52], and other image processing applications [53, 54, 55, 56]. The graphical representation of the optical flow is shown in Fig. 4.7.

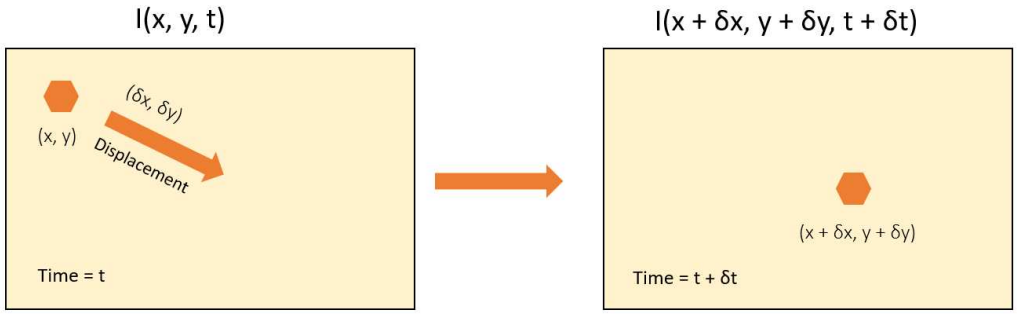


Figure 4.7. *Pixel displacement across two consecutive frames.*

Let us assume that a point is located at (x, y), and the image intensity at this point is $I(x, y, t)$ at time t. The same point moves to its new location $(x+ \delta x, y+ \delta y)$ at $t + \delta t$ time as shown in Fig. 4.7. It is considered that the brightness of an image point remains same over time. This means that the observed brightness of any object in the scene is same over time in the previous and next image frames, which is called the brightness constancy assumption.
The brightness constancy assumption is defined as

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t) \tag{4.9}$$

The motion is assumed to be relatively small, and the image constraint at $I(x, y, t)$ with Taylor expansion is defined as

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x, y, t)+\frac{\partial I}{\partial x}\delta x+\frac{\partial I}{\partial y}\delta y+\frac{\partial I}{\partial t}\delta t+ \text{ higher-order terms} \tag{4.10}$$

Higher-order terms are negligible since the minimum temporal variation is assumed to be $dt \to 0$. Then it follows

$$\frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t = 0 \tag{4.11}$$

By dividing δt, we get

$$\frac{\partial I}{\partial x}\frac{\delta x}{\delta t} + \frac{\partial I}{\partial y}\frac{\delta y}{\delta t} + \frac{\partial I}{\partial t}\frac{\delta t}{\delta t} = 0 \tag{4.12}$$

After simplification we get

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0 \tag{4.13}$$

Where u is the optical flow vector at x-direction, v is the optical flow vector at y-direction, and $\frac{\partial I}{\partial x}$, $\frac{\partial I}{\partial y}$ and $\frac{\partial I}{\partial t}$ are the gradients of the image. Pixel-based motion estimation relies on computing the optical flow vectors (u, v) for each pixel in the image, and that is referred to as the dense optical flow.

The values of the optical flow field are vectors that indicate the direction of movement of pixels. In simple words, these values show where each pixel is in the next frame. As shown in Fig. 4.8, a pseudo-color coding scheme is used to read the pixel shift between consecutive image frames more easily. The saturation of the colors indicates the magnitudes of the vector, while the hue represents the direction of the motion vector.

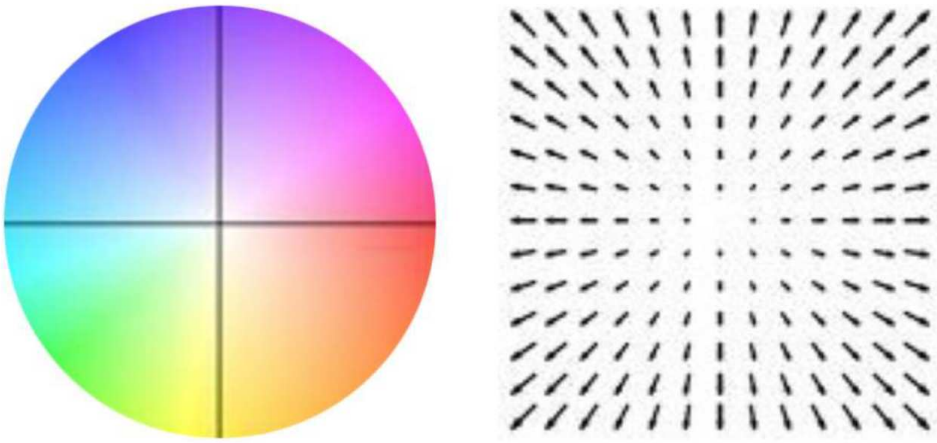


Figure 4.8. *Color-coded representation of vector field and optical flow.*

As mentioned, the equation 4.9 is valid only for relatively small motions and fails for large motions. Due to this limitation, coarse-to-fine approach has been developed to support

large motions, separating the levels by reducing the image size to a coarser resolution. The estimated optical flow at the lowest resolution is given as an input to the next level. Next, the image is warped using the optical flow, then upsampled to the next resolution until reaching the original image resolution, as shown in Fig. 4.9. In each scale the image is warped. This is because a better approximation of image gradients is obtained through warping [57].

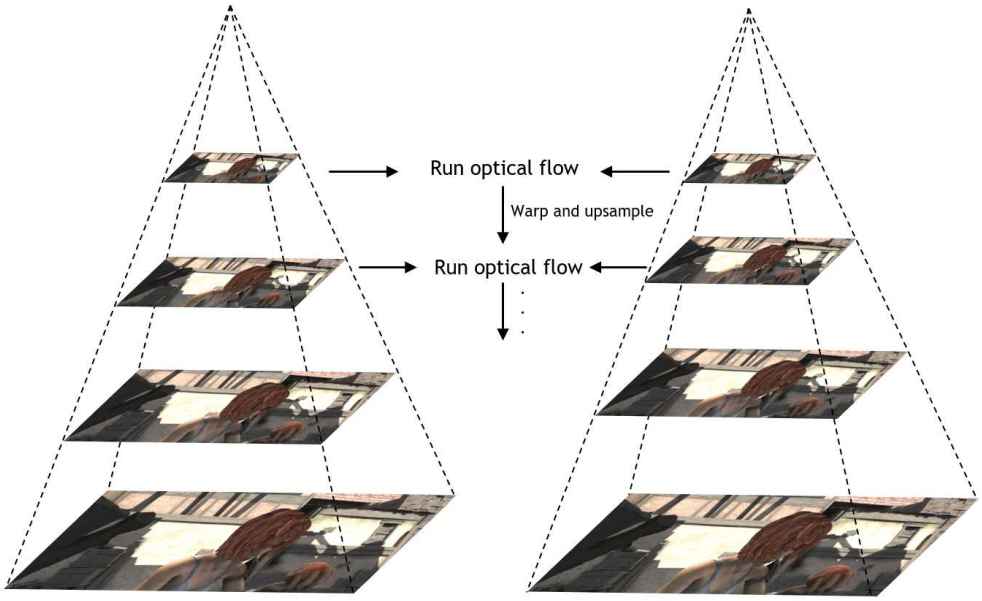


Figure 4.9. *Graphical representation of coarse-to-fine algorithm.*

4.3.2 Dense Optical Flow Estimation Methods

As previously discussed, the optical flow equation is derived by using equation 4.13. However, this equation can not be solved since two unknown variables (u, v) exist in one equation. To solve this problem, some methods have been developed.
Section 4.3.2.a presents conventional dense optical flow methods and Section 4.3.2.b presents deep-learning-based methods used to solve this problem in this thesis.

4.3.2.a Conventional Dense Optical Flow Estimation Methods

There are various implementations of conventional dense optical flow. The methods listed below are the most widely used and publicly available for dense optical flow estimation tasks.

- Lucas-Kanade (LK) method [4]
- Farneback method [6]
- Iterative Lucas-Kanade (ILK) method [7]
- TV-L1 method [8]

- Dense Inverse Search (DIS) method [5]

Lucas-Kanade Method

Lucas-Kanade (LK) method [4] aims to minimize the cost function based on pixel difference. LK method is presented with some prerequisites. These are assumptions of constant brightness and relatively slow motion.

A set equations is required to find the optical flow. LK method is a non-recursive method that assumes a locally constant flow.

In a selected small size of window W x W where W > 1 as shown in Fig. 4.10, a set of equations can be found, assuming that all pixels in the window have the same motion.

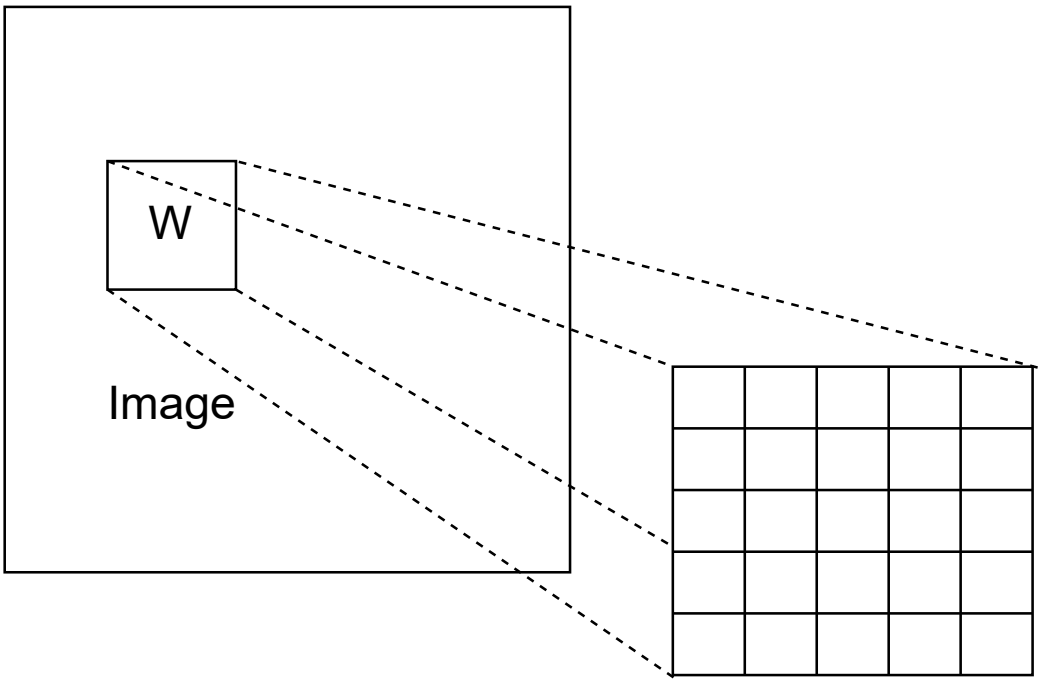


Figure 4.10. *An example of selected window for computing optical flow.*

The following set of equations can be found

$$\begin{aligned} &\mathrm{I_x}\left(\mathrm{k_1}\right)\mathrm{u}+\mathrm{I_y}\left(\mathrm{k_1}\right)\mathrm{v}=-\mathrm{I_t}\left(\mathrm{k_1}\right)\\ &\mathrm{I_x}\left(\mathrm{k_2}\right)\mathrm{u}+\mathrm{I_y}\left(\mathrm{k_2}\right)\mathrm{v}=-\mathrm{I_t}\left(\mathrm{k_2}\right)\\ &\vdots\\ &\mathrm{I_x}\left(\mathrm{k_n}\right)\mathrm{u}+\mathrm{I_y}\left(\mathrm{k_n}\right)\mathrm{v}=-\mathrm{I_t}\left(\mathrm{k_n}\right) \end{aligned} \tag{4.14}$$

where $\mathrm{k_1}$, $\mathrm{k_2}$, ..., $\mathrm{k_n}$ represent the pixels in the window W x W (e.g. n = 25 for a 5x5 window), $\mathrm{I_x}\left(\mathrm{k_i}\right)$, $\mathrm{I_y}\left(\mathrm{k_i}\right)$, $\mathrm{I_t}\left(\mathrm{k_i}\right)$ represent the partial derivatives of image I at point (x, y) at time t, for pixel $\mathrm{k_i}$ at the current time.

Equation 4.14 can be written in an easy-to-read form of $A\mathrm{v}=\mathrm{b}$ matrix.

$$A = \begin{bmatrix} I_x(k_1) & I_y(k_1) \\ I_x(k_2) & I_y(k_2) \\ \vdots & \vdots \\ I_x(k_n) & I_y(k_n) \end{bmatrix} \quad v = \begin{bmatrix} u \\ v \end{bmatrix} \quad b = \begin{bmatrix} -I_t(k_1) \\ -I_t(k_2) \\ \vdots \\ -I_t(k_n) \end{bmatrix} \tag{4.15}$$

To solve the equations, Lucas and Kanade use the "least squares" method for optical flow estimation.

$$A^T A\vec{v} = A^T(-b) \tag{4.16}$$

or,

$$\vec{v} = \left(A^T A\right)^{-1} A^T(-b) \tag{4.17}$$

In matrix form;

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_i I_x(k_i)^2 & \sum_i I_x(k_i) I_y(k_i) \\ \sum_i I_y(k_i) I_x(k_i) & \sum_i I_y(k_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(k_i) I_t(k_i) \\ -\sum_i I_y(k_i) I_t(pk_i) \end{bmatrix} \tag{4.18}$$

Where u and v are optical flow components along x and y directions.
Fig. 4.11d shows a color-coded example of optical flow estimated by Lucas-Kanade method.

Farneback Method

The Farneback method [6] approximates each neighborhood of successive frames using a polynomial expansion transform. This polynomial is a quadratic polynomial. The Lucas-Kanade method is built on linear approximation ($I = k^\top x + z$) and uses only the first order of Taylor expansion. In contrast, Farneback method uses polynomial approximation ($I(x) \cong x^T Dx + k^T x + z$) to estimate flow fields from coefficients (D, k, and z). A weighted least squares is used to estimate the coefficients in accordance with neighboring signal values. D is a 2x2 symmetric matrix that gather information of the even part of the signal, k is a 2x1 matrix that captures information of the odd part of the signal, while z is an unknown scalar [58]. After a series of refinements, dense optical flow is computed. Fig. 4.11e shows a color-coded example of optical flow estimated by Farneback method.

Iterative Lucas-Kanade Method

Iterative Lucas-Kanade (ILK) method proposed a modified scheme of Iterative Warping Scheme (IWS) with better performance since the usual IWS encounters the problems of divergence [7]. IWS is divided into three part: First-order Displaced-Frame-Difference

(DFD) residual approximation, optical flow refinement, and warping of image using optical flow computed at each level [59]. Such an iterative process may be used in a coarse-to-fine multi-resolution strategy as well as at each level of resolution. In [7], they focus on iterative window-based optical flow estimation in case of large motion. Moreover, they state that strict implementation of IWS causes inconsistencies in LK algorithm. So that, they came up with a fast and convergent IWS by using n^{th} levels B-spline separable kernel interpolation [60]. ILK gives better results and lower cost compared to the LK algorithm on artificially generated or real images. Fig. 4.11f shows a color-coded example of optical flow estimated by ILK method.

TV-L1 Method

Zach et al. [61] introduced an optical flow estimation method based on the brightness constancy assumption in 2007 and improved by Sanchez at al. in 2013 [8], aiming to minimize the total variation of a function, including a data term and a regularization term using L1 norm. Total Variation (TV) regularization term is adopted to penalize high variations in optical flow field to obtain smooth optical flow field [61]. The main benefit of this formulation is that TV-L1 method offers more accurate and reliable results in noisy environments than the classical approach of Horn and Schunck [2] while allowing discontinuities in the flow field. TV-L1 method is separated into two modules. The first module is called procedure and it computes the optical flow at a given scale. The second module is the main algorithm that applies the pyramid scheme. The procedure module detects the small motions. [8] claim that there is no need for a pyramid of images when the flow field is below 1 pixel, and if it is larger than 1 pixel, a pyramid level should be used. In other words, TV-L1 uses a pyramidal scheme to overcome large displacement. Fig. 4.11g shows a color-coded example of optical flow estimated by TV-L1 method.

Dense Inverse Search Method

Kroeger et al. [5] came up with a solution for dense optical flow estimation to keep the execution time low and the accuracy high. Dense Inverse Search (DIS) method includes three main modules. The first module is the inverse search for patch correspondences, where the best matching sub-window of WxW pixel in the second frame is found by using gradient descent for a given template patch in the first frame at the location of $a = (x_0, y_0)$ with a size of WxW pixel. The idea is to find a warping vector $w = (u, v)$ to minimize the sum of squared differences (SSD) between the query location and the given template over the sub-window. The second module is the dense displacement field creation through patch aggregation along with multiple scales, where the intermediate dense flow field is computed, and the displacements are smoothed by the dense flow field to provide robustness. Each scale has five steps: criterion of a grid, initialization, inverse search, densification, and variational refinement. The last module is the variational refinement, where the flow field is refined from the coarser to fine level. Fig. 4.11h shows a color-coded example of optical flow estimated by DIS method.

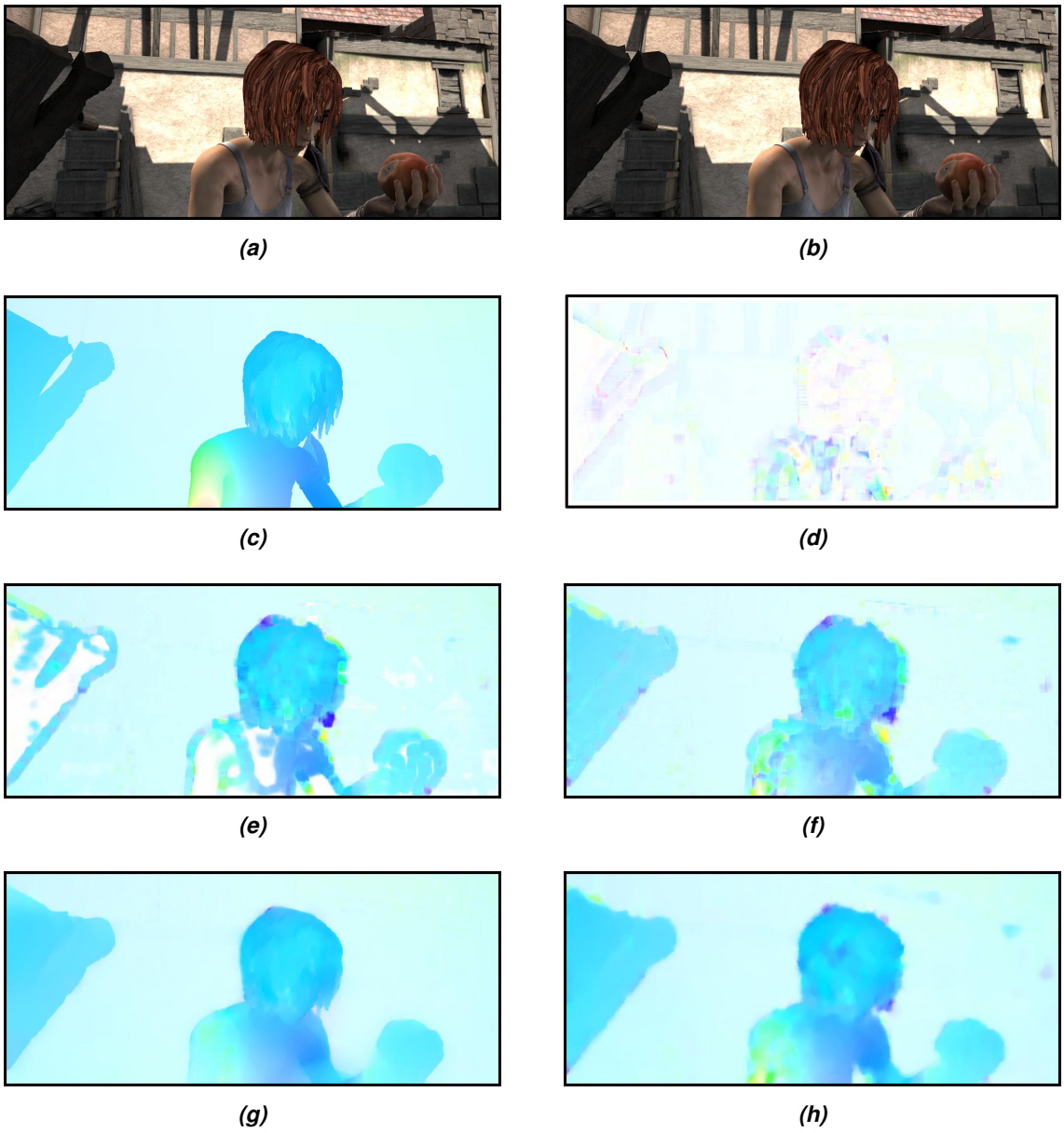


Figure 4.11. *Color-coded representation of estimated optical flow using conventional methods. ((a), (b)) Consecutive frames from Sintel dataset [15]. (c) Ground truth optical flow between given image pair. (d) Estimated optical flow by Lucas-Kanade Method. (e) by Farneback Method. (f) by ILK Method. (g). by TV-L1 Method. (h) by DIS Method.*

4.3.2.b Deep-Learning Based Dense Optical Flow Estimation Methods

Section 4.3.2.b describes five papers written about optical flow estimation [10, 12, 62, 63, 64]. There are several reasons behind the choice of these papers as follows:

- Their script and pre-trained models are publicly available to test [65, 66, 67, 68, 69].
- MPI Sintel benchmark [70] presents the results and ranking on their website. The top 100 are considered.
- PWC-Net[10] is purely based on CNN. PWC-Net is picked to compare with trans-

former based methods[12, 64].

CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume

Sun et al. [10] introduced a CNN-based approach for dense optical flow estimation tasks, which consists of three layers: pyramid layer, warping layer, and cost volume layer, as seen in Fig. 4.12. A pyramid (P in the PWC) downsamples features using convolutional filters in each level of the pyramid. The warping layer (W in the PWC) warps the second frame features onto the first frame by using two times upsampled flow from the next resolution in each pyramid level. The cost volume layer (C in PWC) can be defined as the similarity between the warped and reference image features. In other words, this can be called pixel-wise similarity loss. Fig. 4.17d shows a color-coded example of optical flow estimated by PWC-Net method.

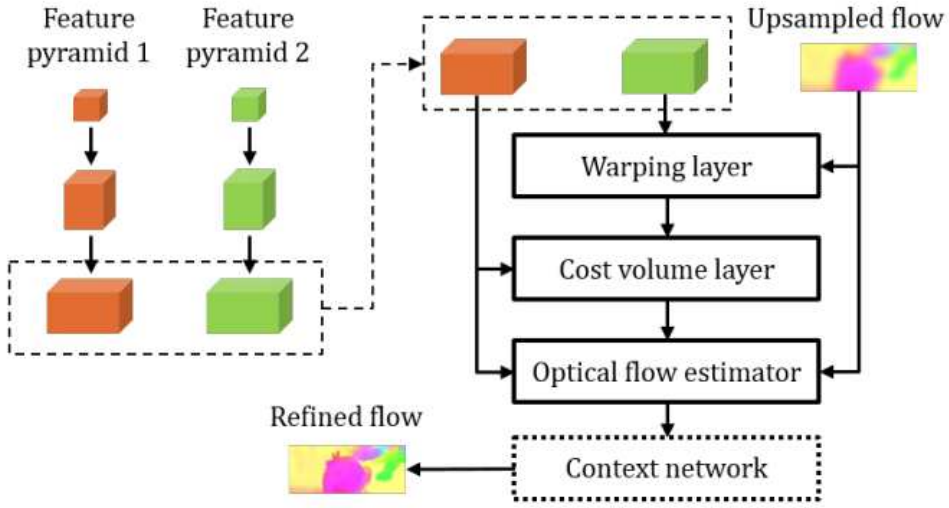


Figure 4.12. *The architecture of PWC-Net obtained from [10].*

Recurrent All Pairs Field Transforms

Teed et al. [62] proposed a deep learning method for dense optical flow estimation task, combining CNN and RNN. The architecture of Recurrent All Pairs Field Transforms (RAFT) is shown in Fig. 4.13. The network architecture consists of three modules: feature extractor module, correlation layer module, and update operator module. The feature extractor module is used to extract significant features using a CNN. The input consists of two successive frames. A CNN with six residual layers extracts the features of these input frames. Correlation layer module is used to construct 4D (H x W x H x W) correlation volume. Correlation volume aims to compute the similarity between a certain part of the reference frame and each part of the next frame in brute force. The update operator module is used to estimate a set of flow estimates (f_1, ..., f_s) from the initial starting point $\mathrm{f}_0 = 0$. After each iteration, the update operator produces a new updated flow to make the prediction more robust in each step by performing inferences iteratively. For instance, if the iteration number is kept small, the computation time is short, and the accuracy is

relatively low. If the iteration number is kept large, the computation time is long, and the accuracy tends to be relatively high. The optical flow results in 1/8 resolution of the initial frame after the iterative update stage. In [62], they used convex upsampling methods to reach the ground truth resolution. The optical flow is upsampled until it reaches the original resolution by taking the original resolution flow in every pixel as a convex combination of limited kernel support of a 3x3 grid of coarse resolution neighbors. Fig. 4.17e shows a color-coded example of optical flow estimated by RAFT method.

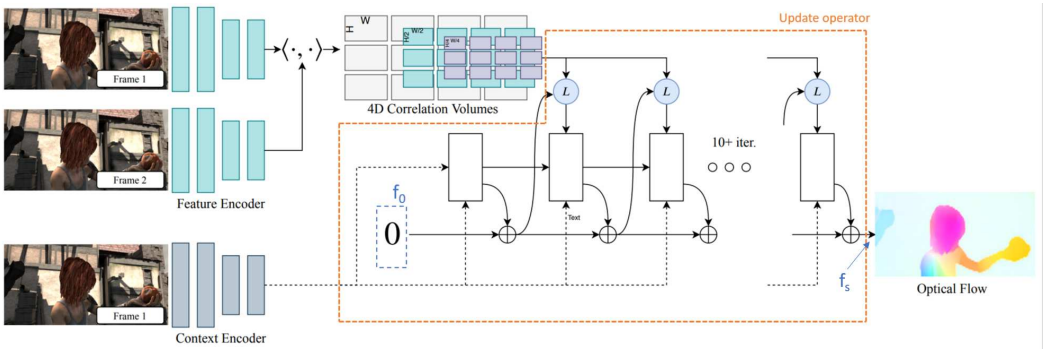


Figure 4.13. *The architecture of RAFT obtained from [62].*

Normalized Convolution Upsampling for Refined Optical Flow Estimation

As previously discussed, optical flow estimation tasks fail for large motions, and coarse-to-fine approach was developed to support large motions. By filtering and resampling images at a lower resolution, a pyramid of coarse to fine is created. Then the image is warped and upsampled to the next resolution. This process is performed until reaching the original resolution of the image. Bilinear interpolation is generally used to refine the image while warping iteratively. As a result of the warping, fine details and some information are usually lost during this process. To restore the fine details, post-processing is needed. Eldesokey et al. [63] proposed an upsampling approach, Normalized Convolution Upsampler (NCUP), to produce the full resolution flow while optical flow CNNs are being trained. Teed et al. [62] used convex upsampling methods to reach the ground truth resolution. This upsampler is about 10% of the whole network with respect to the number of parameters. Eldesokey et al. [63] replaced and tested their upsampler module on RAFT architecture and achieved better results on the Sintel dataset [15] while having 400k fewer parameters than RAFT. The architecture of the RAFT-NCUP is shown in Fig. 4.14. Fig. 4.17f shows a color-coded example of optical flow estimated by RAFT-NCUP method.

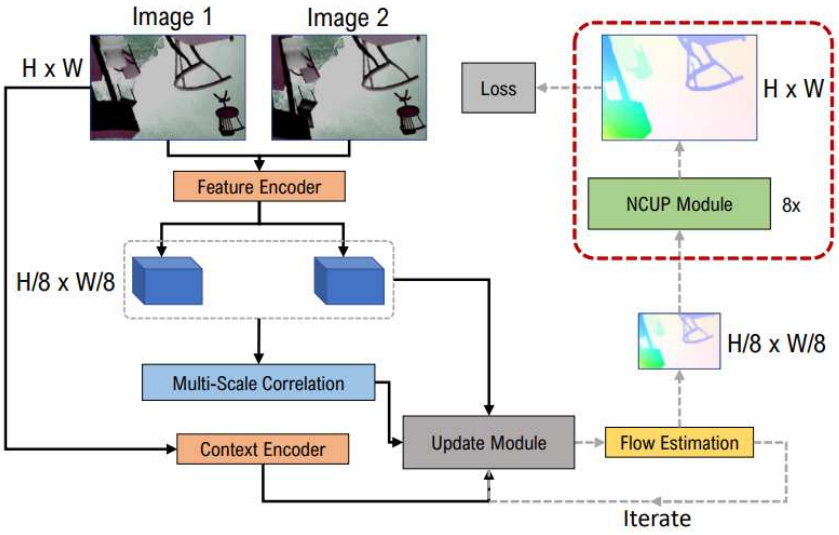


Figure 4.14. *The architecture of RAFT NCUP obtained from [63].*

Global Motion Aggregation

Jiang et al. [12] proposed a new approach to a major challenge: occlusions in optical flow tasks. The problem of occlusions occurs in the images when points become untraceable due to global and local motion inside the scene, which results in unreliable estimates due to the discontinuity [71]. In [12], they claim that, in the case of two frames, the motion in the occluded area can be better estimated by modeling the self-similarities of the image. A transformer-based global motion aggregation module is introduced to find large dependencies among pixels in the reference frame and perform global aggregation on corresponding motion features. In [12], they have shown that the performance of optical flow estimation can be significantly increased in occluded regions without reducing optical flow performance in non-occluded regions. Global Motion Aggregation (GMA) network uses the successful RAFT [62] architecture as a base. Detailed GMA architecture is shown in Fig. 4.15. 2D context features pass through a query projector, and key projector creates a query and a key feature map which is then used to model the self-similarity appearance in the reference frame. Then these feature maps pass through dot product and softmax block to obtain a 4D attention matrix that can capture long-distance dependencies. Separately, 2D motion features pass through the value projector block, which is used to project the 4D correlation volume. The 4D attention matrix is used to collect the features of the value, which are latent representations of motions. Then those features are concatenated to be sent to Gated Recurrent Unit (GRU) to decode the visual context features into the residual flow. Fig. 4.17g shows a color-coded example of optical flow estimated by GMA method.

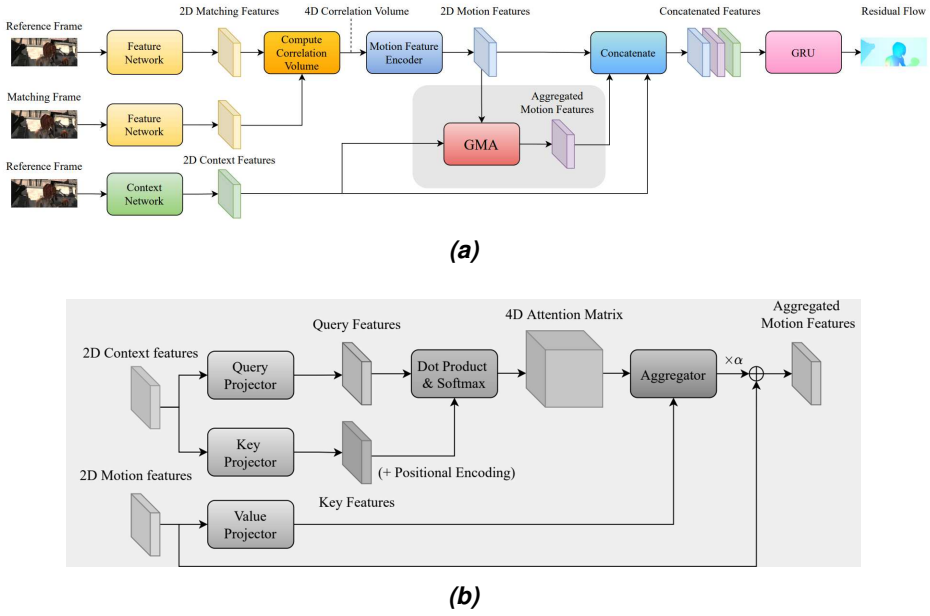
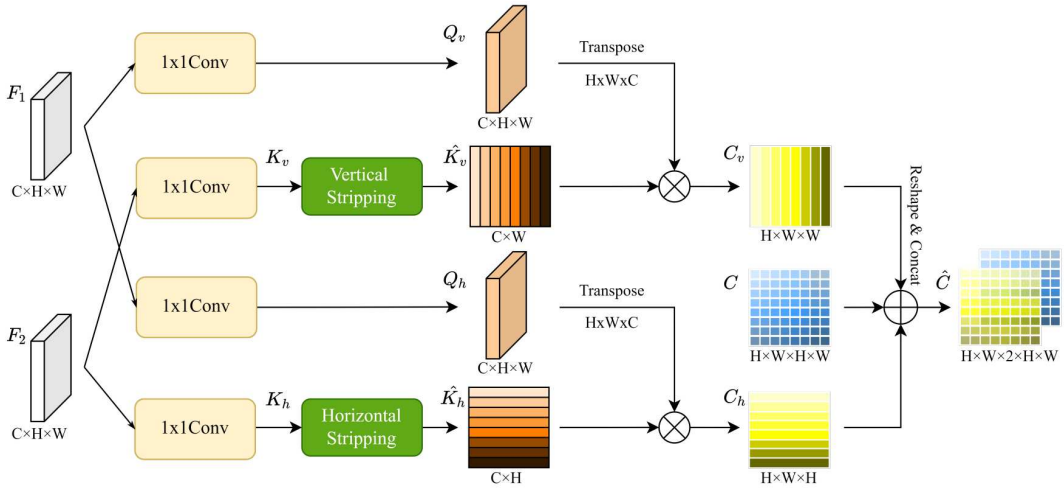


Figure 4.15. *(a), The architecture used in GMA. (b). Details of the GMA module obtained from [12] highlighted gray in (a).*

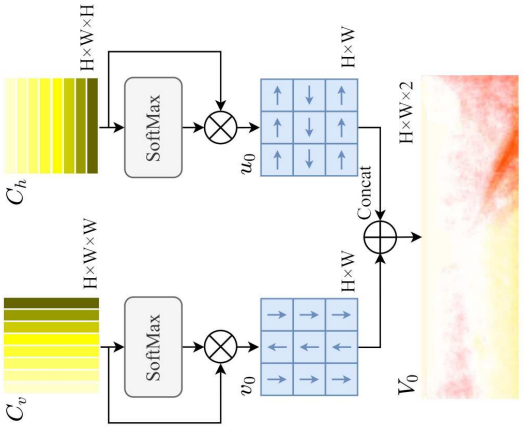
Learning Optical Flow via Cross Strip Correlation (CSFlow)

Shi et al. [64] proposed a new approach for optical flow estimation in the field of autonomous driving. This method consists of two modules. The first module as shown in Fig. 4.16a, is the Cross Strip Correlation (CSC) module which uses a stripping process across the target frame and reference frame to encode global context into correlation volumes to capture large displacement while maintaining high efficiency. The second module as depicted in Fig. 4.16b, is the Correlation Regression Initialization (CRI) module which is used to make maximum use of the global context for initiating optical flow with no additional parameters. In [64], they state that due to the computationally expensive high-level features. Capturing non-local scenes usually requires heavy networks such as Deep Lab [72], and Resnet-101 [73], which are unpractical to use for optical flow estimation tasks. They investigated that the self-attention mechanism can be used for capturing large displacements. However, the calculation of high-level features is still expensive because of the need to compute a relatively huge attention map [73] since the correlation volumes are in the 4D (H x W x H x W) form. To overcome this issue, optical flow can be inherently decomposed into two 1D motion vectors that intersect horizontally and vertically. The whole architecture of CSFlow network is illustrated in Fig. 4.16. The feature maps (F_1, F_2) of two successive frames are obtained by the feature extraction stage. Two 1x1 convolutional layers are used to activate the feature map of the first frame and acquire horizontal and vertical query matrices (Q_v and Q_h). Furthermore, two more 1x1 convolutional layers are used to activate the feature map of the second frame, then $\hat{K}_v$ and $\hat{K}_h$ are obtained by vertical and horizontal striping operations, which are the global key

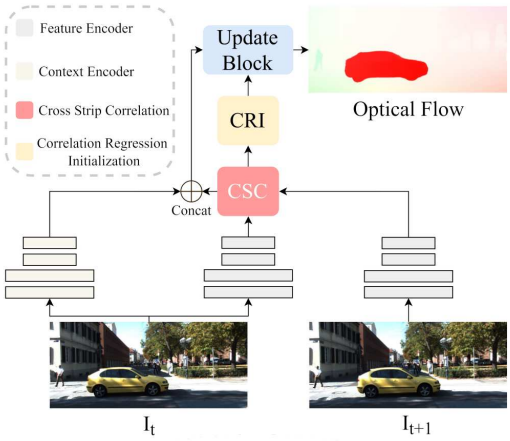
matrices. Then Q_v and Q_h are transposed and apply dot product with $\hat{K}_v$ and $\hat{K}_h$ to get vertical and Horizontal correlation volumes. Lastly, vertical and horizontal correlation volumes are concatenated with the all-pair correlation volume to obtain an aggregated correlation. In order to calculate the optical flow between I_t and I_{t+1}, the aggregated volume $\hat{C}$ and initial optical flow provided by the CRI module then will be used as an input of the update block to compute optical flow as shown in Fig. 4.16c. Fig. 4.17h depicts a color-coded example of optical flow estimated by CSFlow method.



(a) Cross Strip Correlation (CSC) module



(b) Correlation Regression Initialization (CRI) module



(c) Update Block

Figure 4.16. *The architecture of CSFlow obtained from [64].*

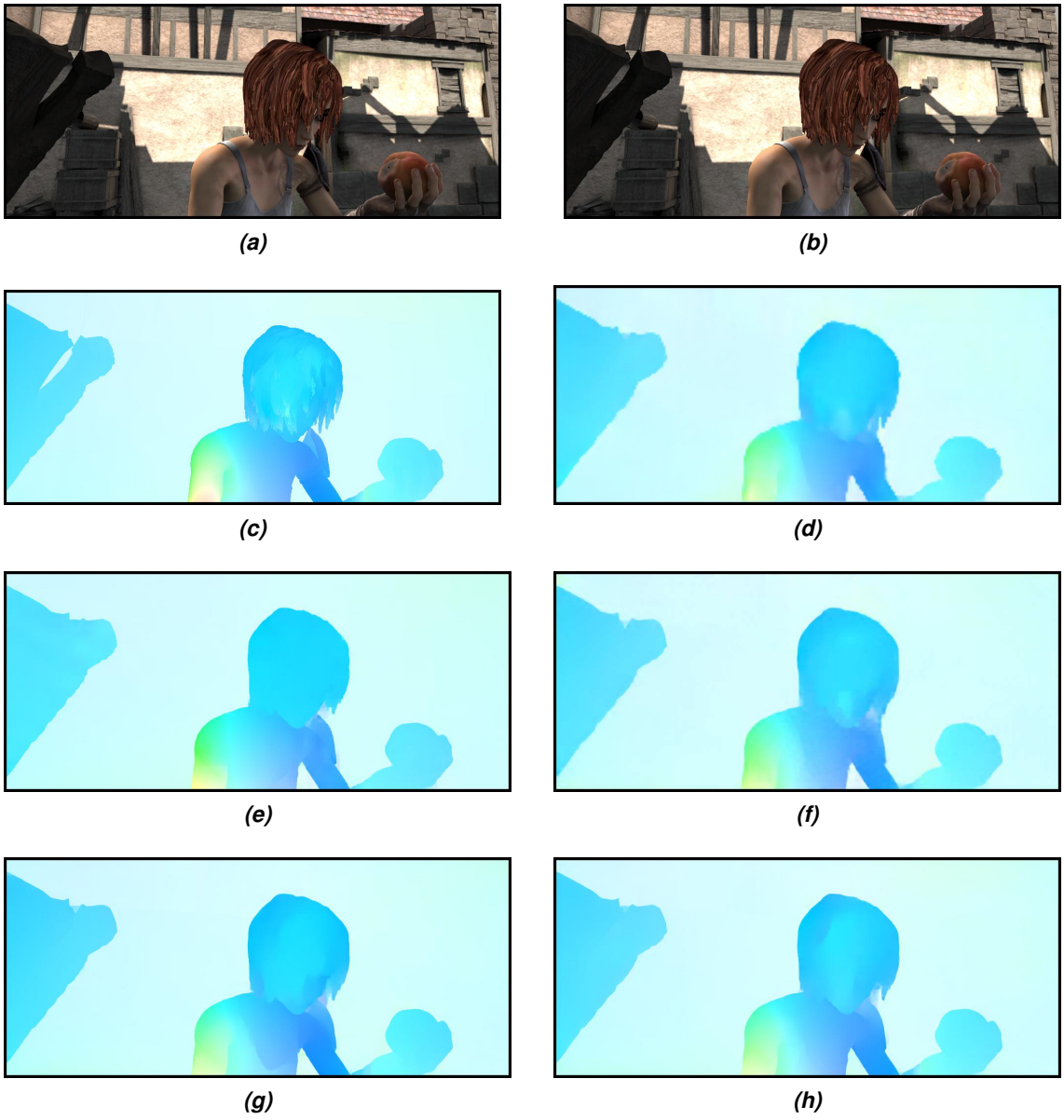


Figure 4.17. *Color-coded representation of estimated optical flow using deep learning methods. ((a), (b)) Consecutive frames from Sintel dataset [15]. (c) Ground truth optical flow between given image pair. (d) Estimated optical flow by PWC-Net method. (e) by RAFT method. (f) by RAFT-NCUP method. (g). by GMA method. (h) by CSFlow method.*

5. IMAGE WARPING AND FUSION

This chapter introduces two main processes on which image alignment and fusion pipeline is built as shown in Fig. 1.3. Section 5.1 describes image warping. Then, Section 5.2 describes the image fusion.

5.1 Image Warping

Image warping is a common technique used in various image processing applications such as image alignment and coarse-fine optical flow [10, 74]. This technique is defined as a geometric transformation that maps all pixels of the destination image to the necessary locations in a source image. A typical image warping process is illustrated in Fig. 5.1.

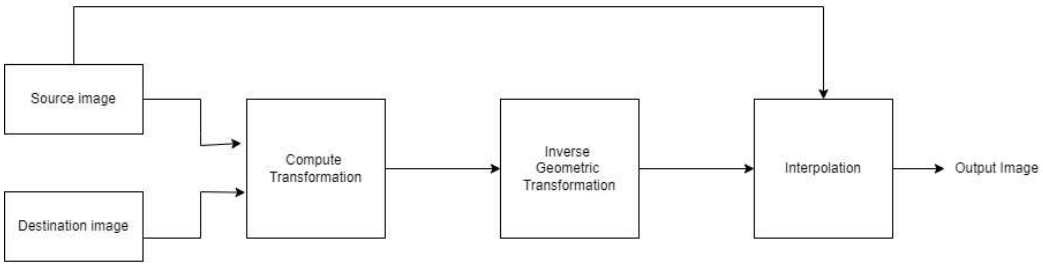


Figure 5.1. *Block diagram of image warping*

Consider two images that need to be aligned by image warping: source and destination images. Then, a transformation matrix is computed such as dense optical flow or homography. Next, the inverse geometric transformation inverts the transformation matrix to map pixels from the destination image to the source image. The transformation matrix and/or its inverse can point to source image coordinates (x, y) with non-integer values. Lastly, the warping algorithm uses various interpolation techniques to produce the pixel intensity values at these integer pixel coordinates. Interpolation methods could differ greatly in complexity and accuracy. However, bilinear and bicubic interpolation are the most commonly used interpolation methods. The example given in Fig. 5.3 shows how to interpolate the non-integer pixel coordinates. The intensity of the closest integer points, red dots, is used to generate the intensity value of the shifted non-integer point, blue dot.

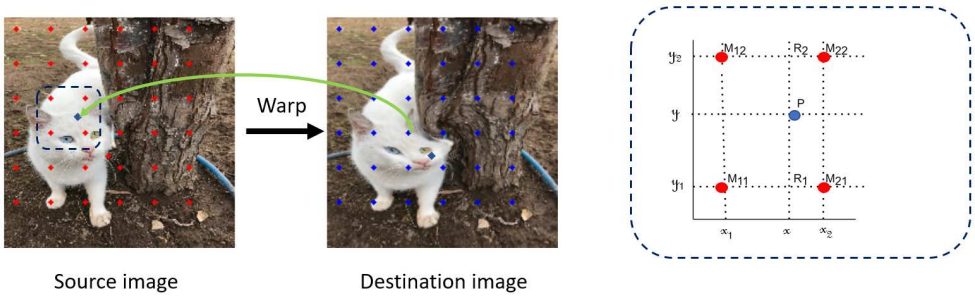


Figure 5.2. *Determining the intensity value of a non-integer pixel (blue dot), using closest points where the intensity values are known (red dots)*

5.2 Image Fusion

Image fusion is the process of gathering important information from multi frames into a single compound frame. The single output frame provides more information than the input frames for visual perception and computer vision processing. Because of this advantage, image fusion techniques are highly important in various applications based on multiple images of the same scene. For example, in medical diagnosis, doctors diagnose the disease of patients by examining multiple modalities of medical images for an accurate diagnosis [75]. Another example of the image fusion is to expand depth of field or dynamic range of a captured image. such as fusing of multi-focus images and multi-exposure images [76, 77].

Fusion technique is used to merge raw image frames captured by the camera sensor to create a single raw frame to be further processed in the ISP pipeline. In this study, a simple fusion technique is performed by averaging the reference image and the warped (aligned) target image. This process is performed for all images to obtain a single image. A simple example of fusion is shown in Fig. 5.3. Fig. 5.3a shows fusion between the reference and target images, while Fig. 5.3b illustrates fusion between the reference and warped target images. It can be seen that fusing images without aligning may cause some ghosting artifacts while ghosting artifacts are mostly removed by aligning target images before fusing.

(a) ***(b)***

Figure 5.3. *An example of image fusion between two consecutive processed raw images. (a) Fusion without warping the target frame. (b) Fusion after warping the target frame.*

6. DATABASES

Datasets are important as they are used for both training and inferencing. The rule of thumb is that the more data used in training, the more accurately trained models are achieved. This chapter describes the datasets used for optical flow estimation tasks. Fig. 6.1 shows the realism of the datasets and the number of image pairs (size) each dataset has.

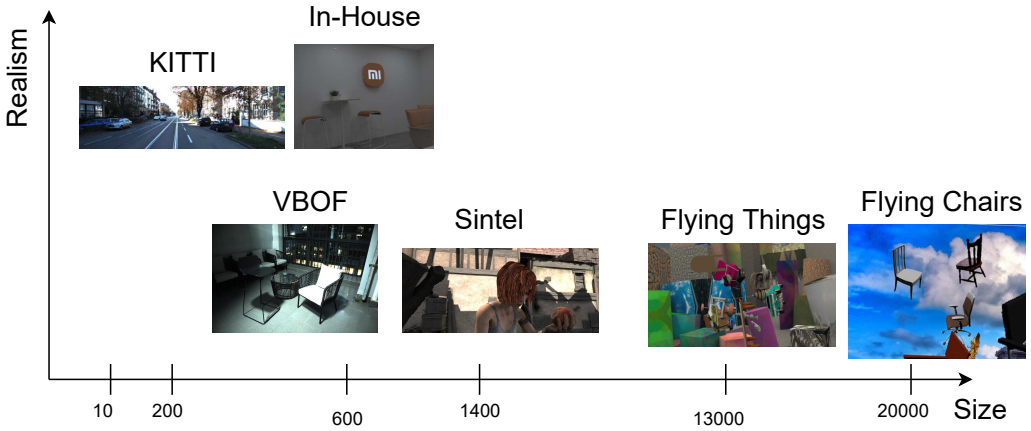


Figure 6.1. *The realism of the datasets with respect to their size.*

Section 6.1 presents the Sintel dataset. Then, the Flying Chairs dataset is described in Section 6.2. Next, Section 6.3 explains the KITTI dataset. After that, Section 6.4 covers the Various Brightness Optical Flow (VBOF) Dataset. Thereafter, Section 6.5 describes the in-house dataset. Finally, the Flying Things dataset is presented in Section 6.6.

6.1 Sintel Dataset

The Sintel dataset [15] is the most commonly used dataset in optical flow tasks for both training and evaluating the optical flow algorithms [9, 10, 62]. It is an open source dataset that has been artificially generated with three different passes: albedo, clean, and final. In albedo pass, images do not contain shading such as reflection and shadow. In clean pass, images have shading but do not contain any image degradation such as motion blur and noise. In final pass, images contain motion blur and atmospheric effects. The Sintel dataset has 35 scenes for each pass, 23 scenes for training, and 12 scenes for testing.

Scenes can differ according to their characteristic, such as large displacements, small displacements, discontinuity, etc. Images are saved in 8-bit PNG format with a 1024 x 436 pixels resolution. A set of example and ground truth images can be seen in Fig. 6.2

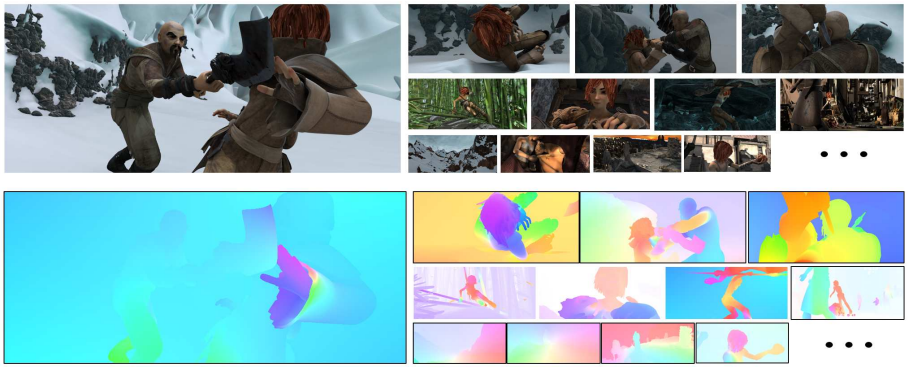


Figure 6.2. *Example images and color-coded flow field (ground truth) from the Sintel dataset.*

6.2 Flying Chairs Dataset

The Flying Chairs dataset was created by the authors of Flow-Net paper[9]. This artificially generated dataset contains 22872 image pairs out of a total of 45744 images. The dataset was created by adding synthetic chairs to selected real-world backgrounds. Image pairs may vary according to their content, such as small motion under daylight conditions, large motion, small motion under low light conditions, occlusion, etc. A set of example and ground-truth images from the Flying Chairs dataset can be seen in Fig. 6.3.

Figure 6.3. *A set of images and color-coded flow field (ground truth) from Flying Chairs dataset.*

6.3 KITTI Dataset

The KITTI dataset is a real-world dataset used for the purpose of autonomous driving applications, object tracking, optical flow, and object detection. There are two versions of the KITTI dataset: KITTI 2012 [78] and KITTI 2015 [79]. In addition to the KITTI 2012, the KITTI 2015 includes dynamic scenes, which makes it more challenging due to massive motion, drastic illumination changes, and occlusions. The ground truth is computed by Light Detection and Ranging (LIDAR). An example image and color-coded flow field (ground truth) from the KITTI dataset can be seen in Fig. 6.4. The ground truth of the KITTI dataset is only approximate due to measurement errors, something intolerable for image alignment. So that, the KITTI dataset is not used in this thesis.

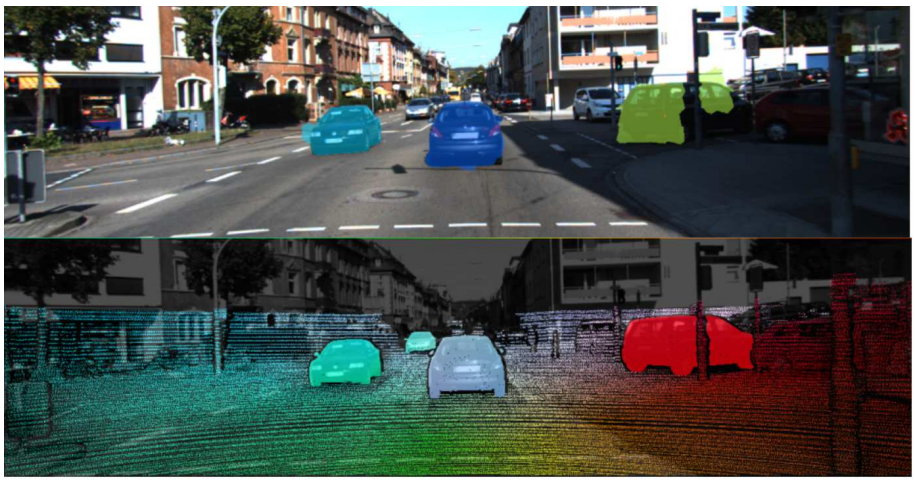


Figure 6.4. *An example image and color-coded flow field (ground truth) from KITTI dataset.*

6.4 Various Brightness Optical Flow Dataset

Various Brightness Optical Flow (VBOF) dataset [80] contains low-light noisy raw images. This dataset contains 598 raw image pairs with various brightness. This dataset was not used in this thesis. However, it can be used as a training dataset in the future to improve the performance of optical flow algorithms under low-light conditions. A set of example and ground-truth images from the VBOF dataset can be seen in Fig. 6.5.

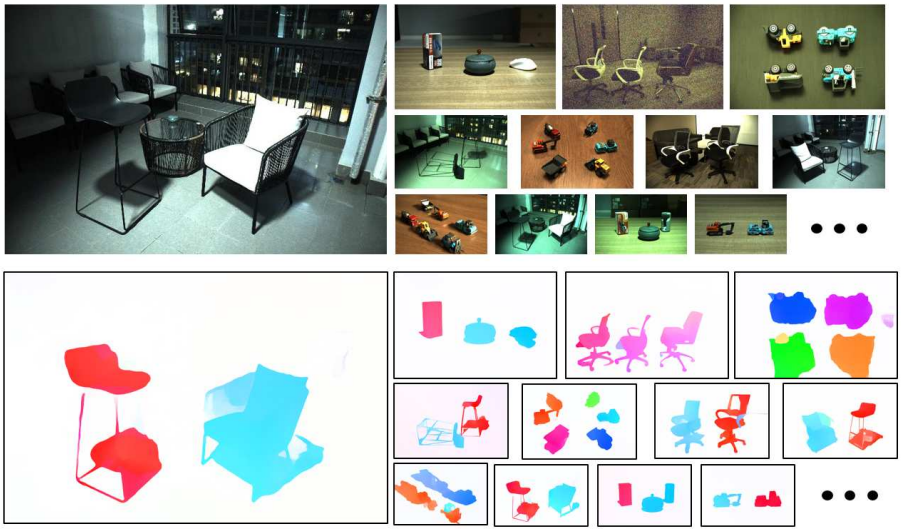


Figure 6.5. *A set of images and color-coded flow field (ground truth) from VBOF dataset.*

6.5 In-House Dataset

In this thesis, the performance of optical flow methods in the raw Bayer domain is targeted. Therefore, the input data is expected to be close to a raw image. However, all publicly available datasets are far from this requirement. The ground truth of the KITTI dataset is only approximate due to measurement errors, something intolerable for image alignment. The Flying Chairs dataset contains the real-world background scene. However, chairs are synthetic and images are fully processed JPEGs. In addition, images in the Sintel dataset are completely synthetic. For these reasons, in-house dataset containing raw images is created to evaluate the performance of optical flow algorithms. The criterion process of the dataset is illustrated in Fig. 6.6.

Section 6.5.1 presents the process of scene capture. Then, Section 6.5.2 describes the process of data simulation.

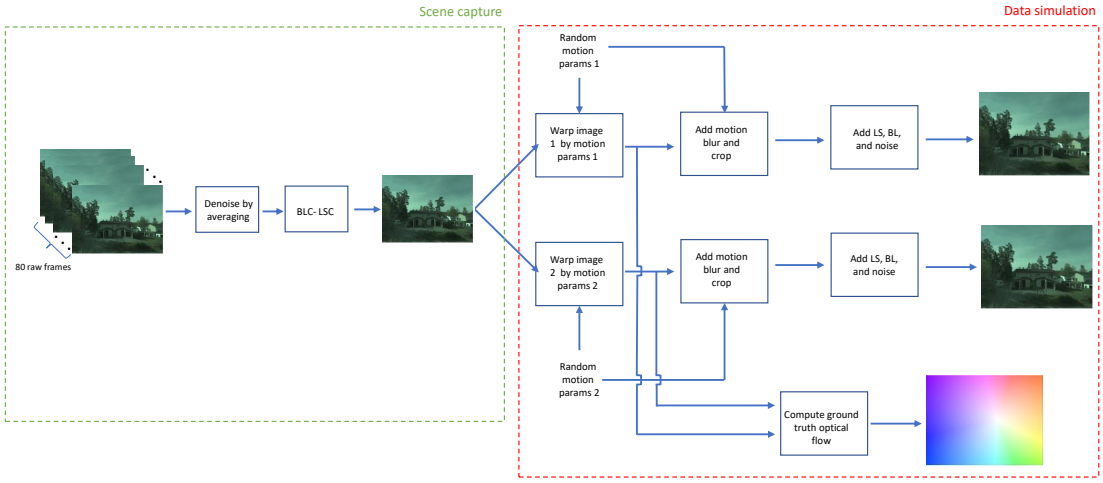


Figure 6.6. *The work flow of scene capture and data simulation.*

6.5.1 Scene Capture

The scene content for the dataset is built on the assumption that the entire scene consists of objects on the same plane. There is no moving object in this dataset. Ten bursts of multi-frame raw image data dump are taken to reduce scene capture noise. Each of the ten dumps in the burst constitutes eight multi-frame raw captures totaling eighty raw frames. Capture noise is reduced by averaging eighty frames to achieve a nearly noiseless scene shot. Then, the noise-reduced capture is further cleaned from the black level and lens shading of the camera module.

6.5.2 Data Simulation

After each scene is cleaned from noise, black level corrected, and lens shading corrected, it is used to create 10 image pairs. Each image pair is obtained from the clean scene image by warping it with two random motions. The type of motion includes the x-axis, y-axis, and z-axis translation and rotation around the z-axis. Once an image pair is created, the relative motion parameters from the reference frame to the target frame are calculated. These parameters are used to construct the ground-truth optical flow.
Then the motion blur is added to both the reference and target frames. For each image pair, an exposure time is drawn from a random uniform distribution ranging between 1 ms and 1/3 of the assumed interval between the simulated captures of the images in the pair. The exposure time is divided equally into 20 time instants and for each time instant, an image is warped from the reference image with motion which is a proportional fraction of relative motion between the reference and the target images in the pair. Finally, twenty differently warped frames are averaged to obtain a single motion-blurred version of the reference and target frames.
Each image pair is degraded with lens shading, add black level, and noise. Then it is saved in 16-bit PNG format. A set of images from in-house dataset is shown in Fig. 6.7.

Figure 6.7. *A set of images from in-house dataset.*

6.6 Flying Things Dataset

Flying Things dataset contains 13033 image pairs out of a total of 26066 images which consists of random objects flying along randomized 3D trajectories [81]. This dataset is used in pre-trained models provided by [10, 12, 62, 63, 64]. Example images and ground truth images from Flying Things dataset is shown in Fig. 6.8.

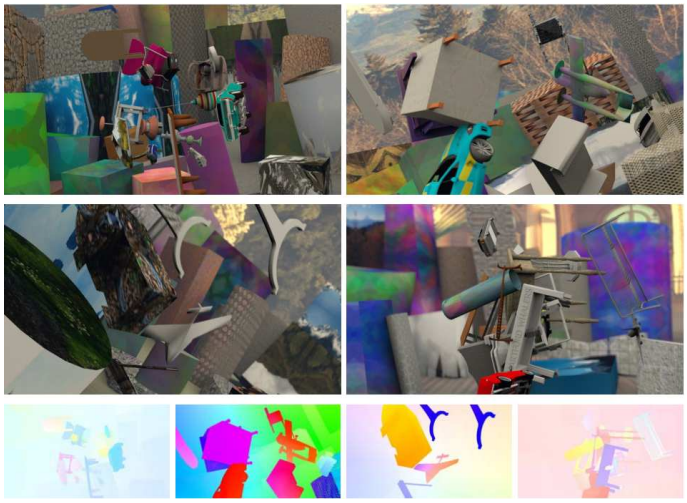


Figure 6.8. *Example images from Flying Things dataset and ground truth images obtained from [81].*

7. EVALUATION METRICS

This chapter introduces the evaluation metrics used to assess the quality of the optical flow estimation and image fusion. Section 7.1 presents the evaluation metrics used for the optical flow evaluation. Then, the evaluation metrics used for image fusion assessment are discussed in Section 7.2.

7.1 Evaluation Metrics for Optical Flow Estimation

Section 7.1 introduces the evaluation metrics used for the optical flow evaluation: Endpoint Error (EPE), Angular Error (AE), and statistical error metrics.

7.1.1 Endpoint Error

Endpoint Error (EPE) is the most commonly used metric to assess the performance of the optical flow by comparing the ground truth optical flow vector with an estimated optical flow vector for each pixel in the image pair. EPE is computed using equation 7.1.

$$EPE = \sum_{i=1}^{N} \sqrt{\left(u_i - u_i^{gt}\right)^2 + \left(v_i - v_i^{gt}\right)^2} \tag{7.1}$$

Where u and v are the (x, y) pixel location in estimated optical flow, while u^{gt} , v^{gt} are the (x, y) pixel location in ground truth optical flow.

7.1.2 Angular Error

Angular Error (AE) is another metric used to assess the performance of the optical flow [82]. It is described as the angle between the ground truth optical flow vector field and the estimated optical flow vector field as shown in Fig. 7.1. AE is computed using equation 7.2. AE suffers from the problem that would tolerate large pixel error for faster moving parts of the scene. Thus it has not been used in this thesis.

$$AE = \sum_{i=1}^{N} \arccos\left(\frac{u_i u_i^{gt} + v_i v_i^{gt} + 1}{\sqrt{u_i^2 + v_i^2 + 1}\sqrt{u_i^{2,gt} + v_i^{2,gt} + 1}}\right) \tag{7.2}$$

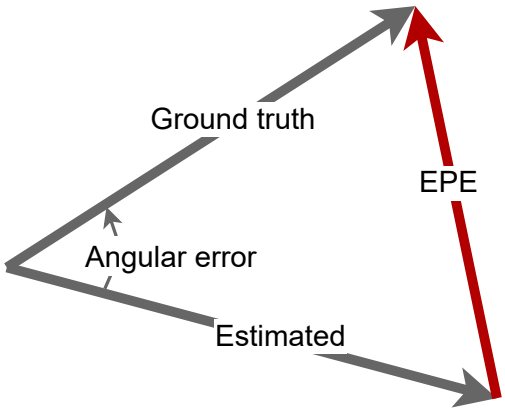


Figure 7.1. *Diagrammatic representation of EPE and AE.*

7.1.3 Statistical Error Metrics

Categorizing by Speed

For example, a well-predicted optical flow with few large errors may have the same error rate as an optical flow that is slightly but consistently inaccurate in all parts of the image. To distinguish that, image pixels can be divided into different categories depending on the speed of motion and the average number of EPEs associated with each category.

- s10: average EPE for all pixels with a small motion speed of less than 10 pixels per frame.
- s10-40: average EPE for all pixels with medium motion speed between 10 and 40 pixels per frame.
- s40+: average EPE for all pixels with a large motion speed greater than 40 pixels per frame.

where s is the speed of the motion between consecutive frames, and it can be computed using equation 7.3, where u and v are the ground truth optical flow vectors.

$$s = \sqrt{u(x,y)^2 + v(x,y)^2} \tag{7.3}$$

Categorizing by Error

The metrics given above may be insufficient in measuring performance of the optical flow. Especially algorithms like Multi-Frame Noise Reduction (MFNR) is very sensitive to even the smallest misalignment. Therefore, a new metric is created to support algorithms like MFNR and assesses the quality of the optical flow. Three categories are created according to the EPE value, similar to the speed calculation.

- e0-1: percentage of pixel error with EPE <= 1 pixel.
- e1-5: percentage of pixel error between 1 and 5 pixels.

- e5+: percentage of pixel error greater than 5 pixels.

Grayscale absolute error EPE image is given in Fig. 7.2 is computed by subtracting each component (u,v) of ground truth optical flow and estimated optical flow and then stack them in sequence depth wise. It ranges between $[0, 1]$. The white parts in the image show the correctly predicted motion, while the black parts in the image indicate the parts that were not predicted correctly.

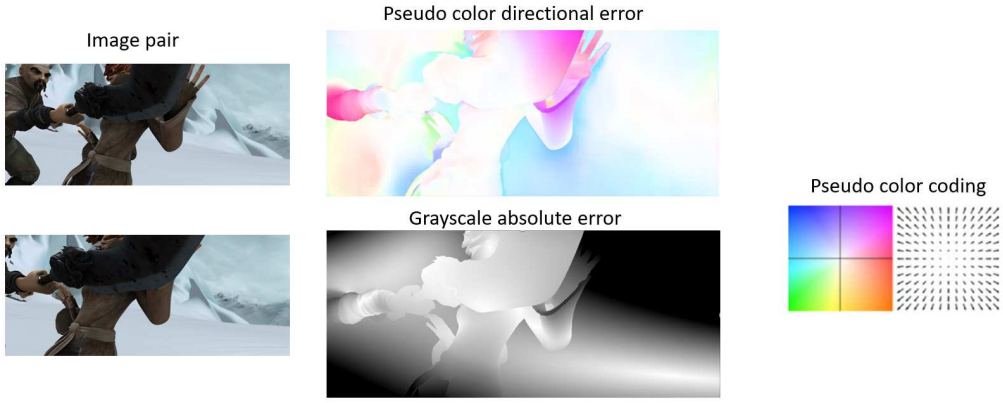


Figure 7.2. *Color coded directional error and grayscale absolute error.*

7.2 Evaluation Metrics for Image Fusion

Section 7.2 describes the evaluation metrics used for assessing the quaility of fused images: Peak signal to Noise Ratio (PSNR) [83], Structural Similarity Index Measurement (SSIM) [84], Spectral Angle Mapper (SAM) [85], Universal Quality Image Index (UQI) [86], and Correlation Coefficient (CC) [87].

7.2.1 Peak Signal to Noise Ratio

Peak Signal to Noise Ratio (PSNR) is the most commonly used metric for measuring the quality between the reference and fused images. The PSNR of the reference frame and fused frame are compared. The PSNR criterion used for comparison is defined as in Equation 7.4

$$\mathrm{PSNR} = 10\log_{10}\frac{255^2 \mathrm{x} \mathrm{H} \mathrm{x} W}{\sum_{i=1}^{H}\sum_{j=1}^{W}(\mathrm{RF}_{ij} - \mathrm{FF}_{ij})^2} \tag{7.4}$$

where RF and FF are the pixel intensities of the reference and fused frames, H, W represent the height and the width of the frames, and 255 indicates the saturation point of 8-bits image. That value varies depending on bit depth.
The higher the value of PSNR indicates higher quality.

7.2.2 Structural Similarity Index Measurement

Structural Similarity Index Measurement (SSIM) is used to compare the local patterns of the pixels intensities between the reference and fused images. SSIM ranges between 0 and 1, where 1 indicates a perfect similarity. The SSIM is obtained by equation 7.5.

$$\mathrm{SSIM}(RF, FF) = \frac{(2\mu_{RF}\mu_{FF} + c_1)\,(2\sigma_{RFFF} + c_2)}{(\mu_{RF}^2 + \mu_{FF}^2 + c_1)\,(\sigma_{RF}^2 + \sigma_{FF}^2 + c_2)} \tag{7.5}$$

where RF and FF are the pixel intensities of the reference and fused frames, μ_{RF}, μ_{WF} are the average of pixel densities, σ_{RF} and σ_{WF} are the standard deviations, σ_{RF}^2 and σ_{FF}^2 are the covariance of the reference and fused frames respectively.

7.2.3 Spectral Angle Mapper

Spectral Angle Mapper (SAM) calculates the pixel-wise spectral angle between the reference and fused images. Lower value of SAM denotes higher similarity. The best similarity is obtained when the SAM value is zero, which indicates the absence of spectral distortion. SAM value is computed by using equation 7.6

$$\mathrm{SAM}(RF, FF) = \cos^{-1}\left(\frac{\sum_{i=1}^{nb} FF_i RF_i}{\sqrt{\sum_{i=1}^{nb} FF_i^2}\sqrt{\sum_{i=1}^{nb} RF_i^2}}\right) \tag{7.6}$$

where RF and FF are the pixel intensities of the reference and fused frames, and nb is the number of bins in the frames.

7.2.4 Universal Quality Image Index

Universal Quality Image Index (UQI) computes the amount of relevant data from reference frame into fused frame. The value of UQI ranges between -1 and +1, where the value of +1 represents the reference and fused frames are similar. The UQI is obtained by equation 7.7

$$UQI = \frac{4\sigma_{RFFF}\mu RF\mu FF}{(\sigma_{RF}^2 + \sigma_{FF}^2)\,[(\mu RF)^2 + (\mu FF)^2]} \tag{7.7}$$

where μRF and μFF are the mean values, σ^2_{RF} and σ^2_{FF} are the variance of reference frame and fused frame respectively.

7.2.5 Correlation Coefficient

Correlation Coefficient (CC) is used to evaluate the similarity of the spectral features of the reference and fused frames. CC value of 1 is obtained when the reference and fused frames are the same. CC value is computed by using equation 7.8

$$CC = \frac{\Sigma\left[(RF - \mu RF)\,(FF - \mu FF)\right]}{\sqrt{\left[\Sigma\,(RF - \mu RF)^2\,\Sigma\,(FF - \mu FF)^2\right]}} \tag{7.8}$$

where μRF and μFF are the mean value of the reference and fused frames respectively.

8. RESULTS

The quantitative and visual results of optical flow methods are presented in Section 8.1. Then, Section 8.2 presents the quantitative and visual results with fused images.

8.1 Quantitative and Visual Results of Optical Flow Estimation Methods

In this thesis, ten optical flow methods are tested. These methods are as follows:

- Conventional Methods
  - SIFT + FLANN
  - Farneback [6]
  - Iterative Lucas Kanade (ILK) [7]
  - TV-L1 [8]
  - Dense Inverse Search (DIS) [5]
- Deep Learning-Based Methods
  - Pyramid, Warping, and Cost Volume Network (PWC-Net) [10]
  - Recurrent All Pairs Field Transforms (RAFT) [62]
  - RAFT method with Normalized Convolution UPsampling (RAFT-NCUP) [63]
  - Global Motion Aggregation (GMA) [12]
  - Cross Strip Correlation (CSFlow) [12]

Section 8.1.1 presents the training procedure of each deep learning-based method. Then, Sections 8.1.2, 8.1.3, and 8.1.4 report the quantitative and visual results of optical flow estimation on the Sintel [15], Flying Chairs [9], and in-house datasets, respectively.

8.1.1 The Training Procedure of Deep Learning Based Methods

The pre-trained model of each deep learning-based method is publicly available [65, 66, 67, 68, 69]. These pre-trained models are used to compare the results on different datasets. The training procedures are provided in Table 8.1. For instance, The GMA

network was first pre-trained on the Flying Chairs dataset for 120k iterations, where the learning rate was 0.00025, the batch size was 8. Then, the model was pre-trained on the Flying Things dataset for 120k iterations, where the learning rate was 0.000125, the batch size was 6. The PWC-Net model was pre-trained on a mix of the Flying Chairs and half resolution of the Flying Things dataset.
Results obtained in this thesis are based on two pre-trained models: C + T and C + $\mathrm{T}_{\mathrm{half}}$. C + T refers to the results pre-trained on the Flying chairs and Flying Things datasets, while C + $\mathrm{T}_{\mathrm{half}}$ refers to the results pre-trained on the Flying Chairs and half resolution of the Flying Things datasets. C + T pre-trained model is available for GMA, CSFlow, RAFT, and RAFT NCUP, while C + $\mathrm{T}_{\mathrm{half}}$ pre-trained model is available only for PWC-Net.

Table 8.1. *Training procedures of deep learning methods.*

Method	Flying Chairs			Flying Things		
	Number of epoch	Learning rate	Batch size	Number of epoch	Learning rate	Batch size
GMA	120k	0.00025	8	120k	0.000125	6
CSFlow	150k	0.0004	10	150k	0.000125	6
RAFT	—	0.0004	6	100k	—	12
RAFT NCUP	—	0.0004	6	100k	—	12

8.1.2 Results on the Sintel Dataset

As mentioned in Section 6.1, the Sintel dataset consists of 35 scenes, 23 for training and 12 for testing. Furthermore, the Sintel dataset provides three different passes. Since low light images usually suffer from motion blur, results are presented for the final pass, which includes motion blur in the images. However, quantitative results for the blur-free clean pass can be seen in the appendix section for both conventional and deep learning methods. No experiment has been conducted on the albedo pass, as there are no image degradations in this pass.

The percent of image pixels according to the speed classification is given in Table 8.2. For example, in $\mathrm{Alley\_1}$, 95.6 % of the motion speed is smaller than 10 pixels. 4.2 % of the motion speed is between 10 and 40 pixels, while only 0.02 % of the motion speed is above 40 pixels.

The final pass results for conventional methods and deep learning-based methods are shown in Fig. 8.1 and 8.2, respectively. These figures show the Average Endpoint Error (AEPE) for each scene, AEPE for different speed categories, and error percentage for different categories. The x-axes indicate the name of each scene, while the y-axis is the magnitude of AEPE and is limited between 0-100 for each plot.

The AEPE is computed for each scene using equation 8.1, where N is the number of image pair in the scene.

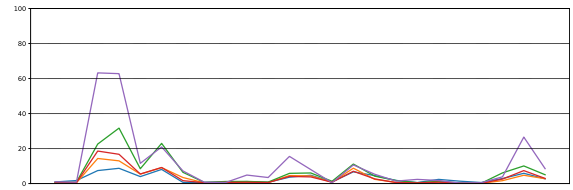
$$\mathrm{AEPE} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{\left(u_i - u_i^{gt}\right)^2 + \left(v_i - v_i^{gt}\right)^2} \tag{8.1}$$

For speed categories, firstly, ground truth optical flow data is used to compute per-pixel shifts between the first and second images, which refers to the speed or displacement between consecutive images. Then, EPE is computed for regions where the speed is between 0-10, 10-40, and 40+ pixels. Additionally, the AEPE values were analyzed separately for regions categorized according to the EPE falling between 0-1, 1-5, and greater than 5 pixels. Lastly, Averaged Error refers to the average EPE of all scenes.

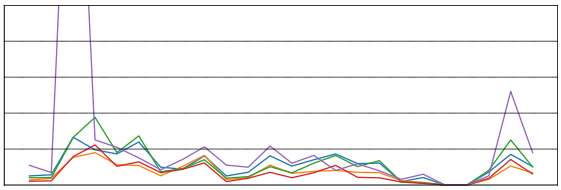
Image pairs, ground truth, and color-coded results of estimated optical flow for four image pairs are illustrated in Fig. 8.3 and 8.4 for conventional and deep learning-based methods, respectively. From left-most to right-most, columns correspond to images for small, medium, large motions, and scene with discontinuity, where the region indicated by the red rectangle. This categorization done by calculating the average speed in the scene. Image pairs, ground truth, and, grayscale absolute error results of estimated optical flow for conventional methods and deep learning-based methods are illustrated in Fig. 8.5 and 8.6 respectively. Furthermore, quantitative results of the AEPE, AEPE in different speed categories and percentage of error are given in Table 8.3 and 8.4 for conventional and deep learning-based methods, respectively.

Table 8.2. *The percentage of image pixels according to the speed classification, where s refers to the speed of the motion.*

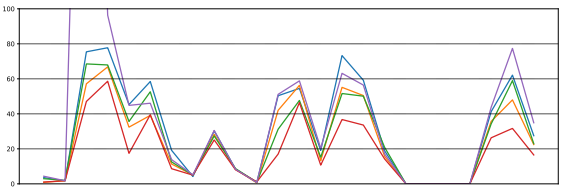
Scene Name	s0-10 (%)	s10-40 (%)	s40+ (%)
Alley_1	95.60	4.20	0.02
Alley_2	78.00	21.60	0.04
Ambush_2	4.00	27.30	68.70
Ambush_4	29.60	47.10	23.30
Ambush_5	52.30	30.10	17.60
Ambush_6	5.00	58.20	36.80
Ambush_7	89.10	7.20	3.70
Bamboo_1	99.10	0.70	0.20
Bamboo_2	96.00	2.05	1.95
Bandage_1	89.00	10.00	1.00
Bandage_2	94.70	5.20	0.10
Cave_2	9.80	43.70	46.50
Cave_4	35.10	60.70	4.20
Market_2	89.90	9.30	0.80
Market_5	10.60	59.60	29.80
Market_6	34.60	55.70	9.70
Mountain_1	87.90	11.60	0.50
Shaman_2	99.80	0.19	0.01
Shaman_3	99.22	0.77	0.01
Sleeping_1	100.00	0.00	0.00
Sleeping_2	100.00	0.00	0.00
Temple_2	60.30	35.40	4.30
Temple_3	26.50	39.60	33.90



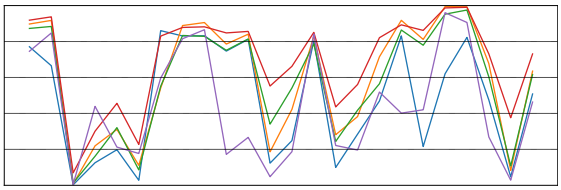
(a) AEPE for all pixels with a small motion speed of less than 10 pixels per frame.



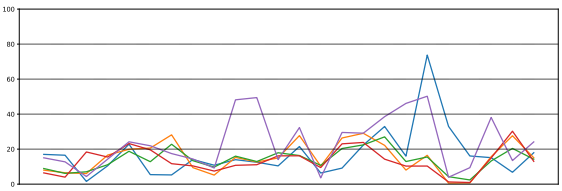
(b) With medium motion speed between 10 and 40 pixels per frame.



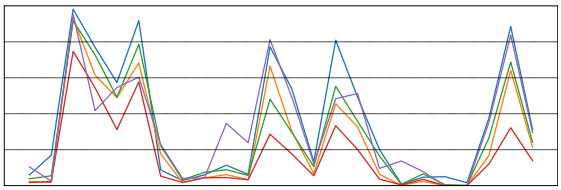
(c) With a large motion speed greater than 40 pixels.



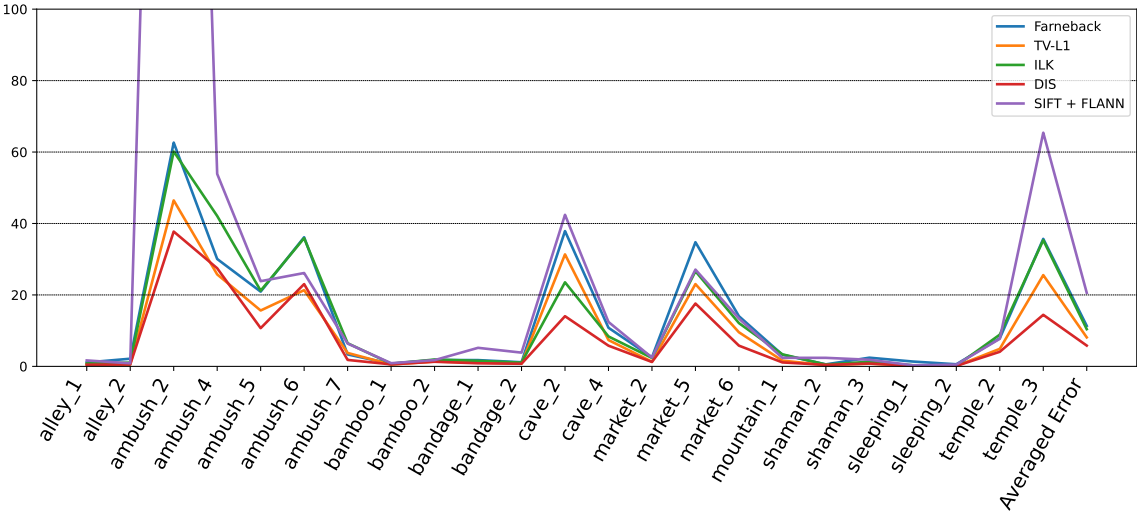
(d) The percentage of pixel error equal or smaller than 1 pixel



(e) The percentage of pixel error between 1 and 5 pixels.

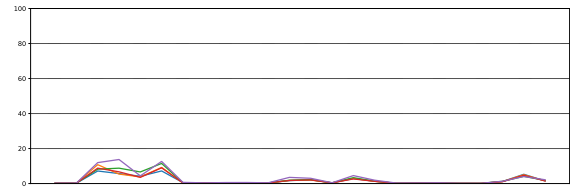


(f) The percentage of pixel error greater than 5 pixels.

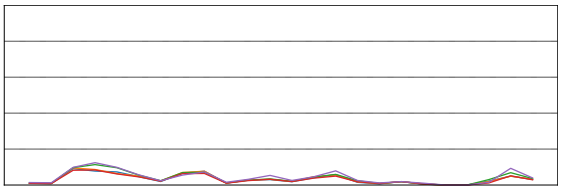


(g) The AEPE of all scenes.

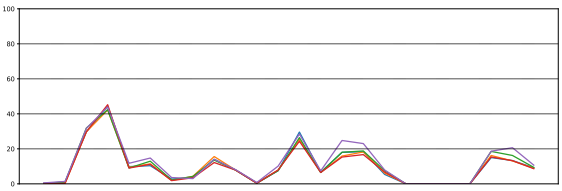
Figure 8.1. *The error plots for conventional optical flow methods.*



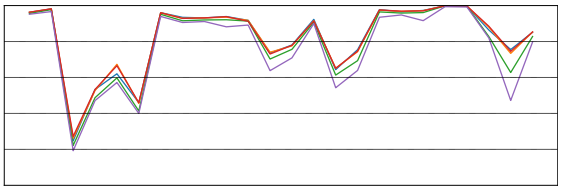
(a) AEPE for all pixels with a small motion speed of less than 10 pixels per frame.



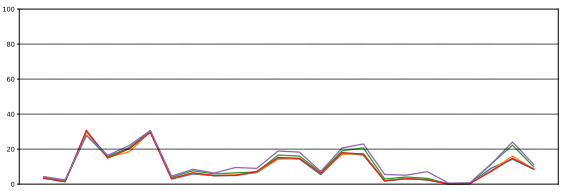
(b) With medium motion speed between 10 and 40 pixels per frame.



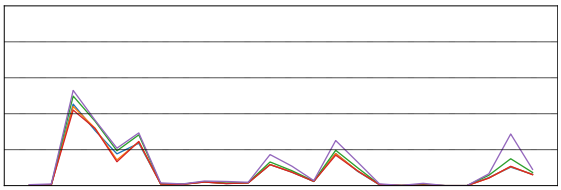
(c) With a large motion speed greater than 40 pixels.



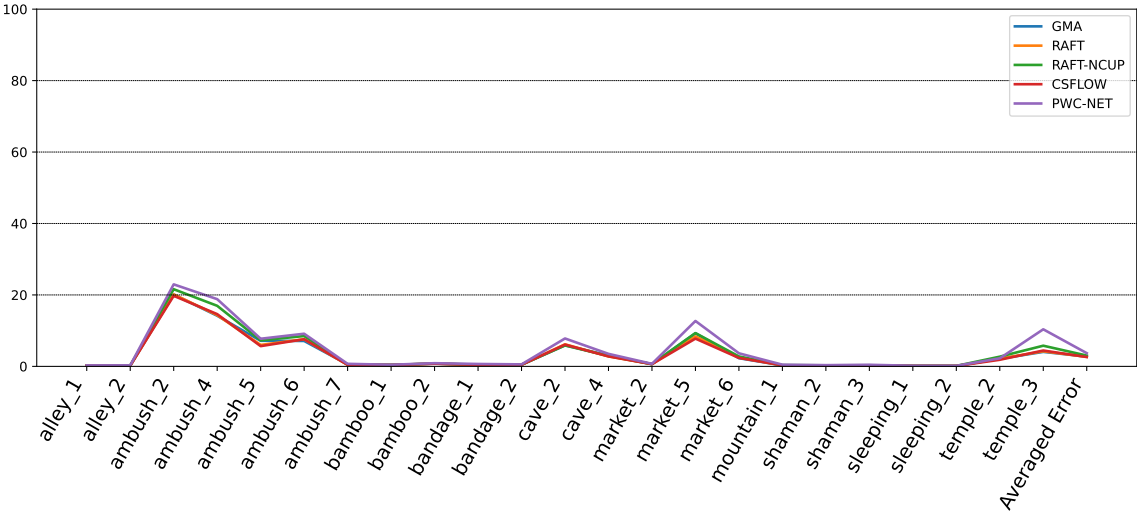
(d) The percentage of pixel error equal or smaller than 1 pixel.



(e) The percentage of pixel error between 1 and 5 pixels.



(f) The percentage of pixel error greater than 5 pixels.



(g) The AEPE of all scenes.

Figure 8.2. *The error plots for deep learning-based optical flow methods.*

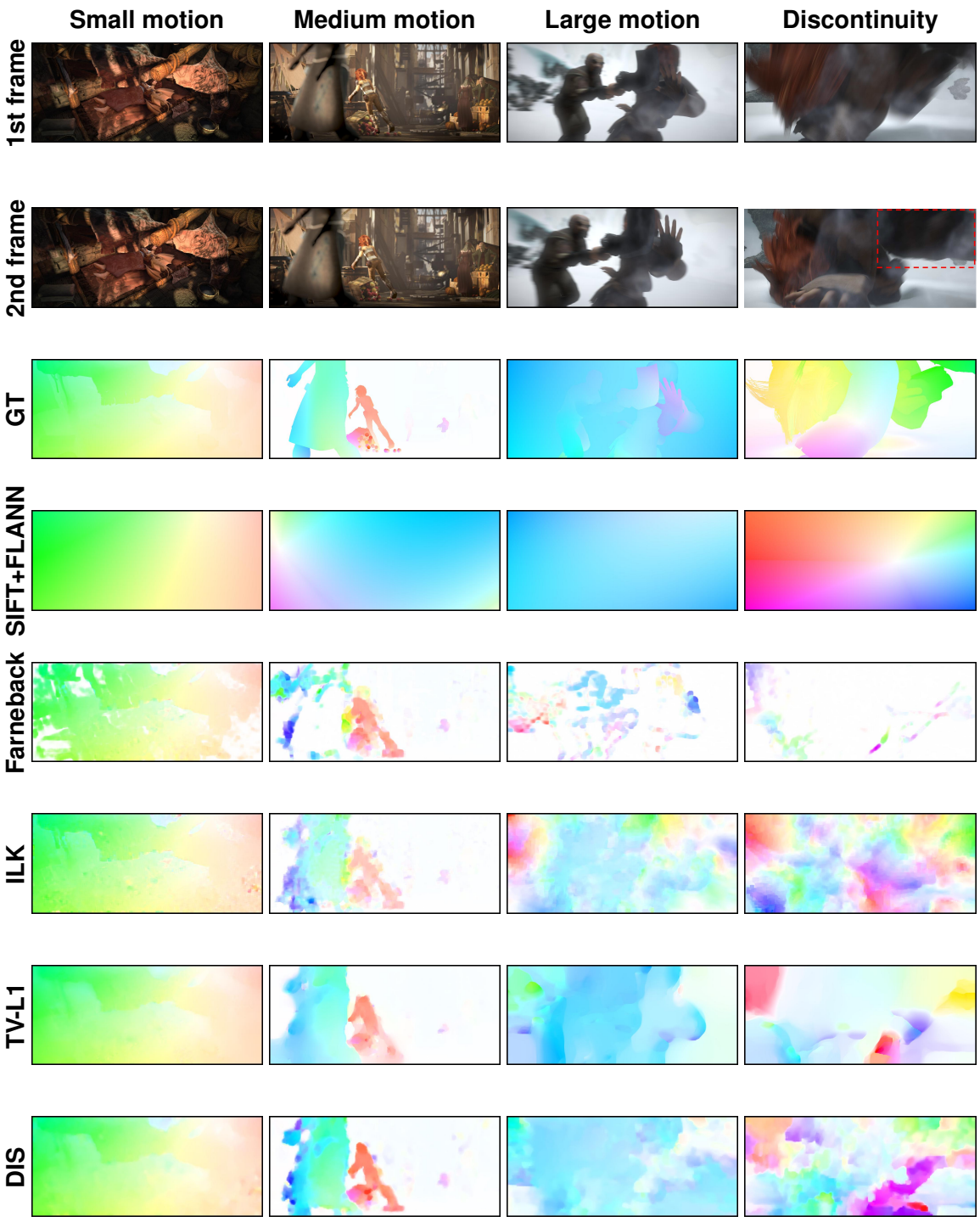


Figure 8.3. *Color-coded results of optical flow estimated by conventional methods for four different scenarios: small, medium, large motion, and discontinuity.*

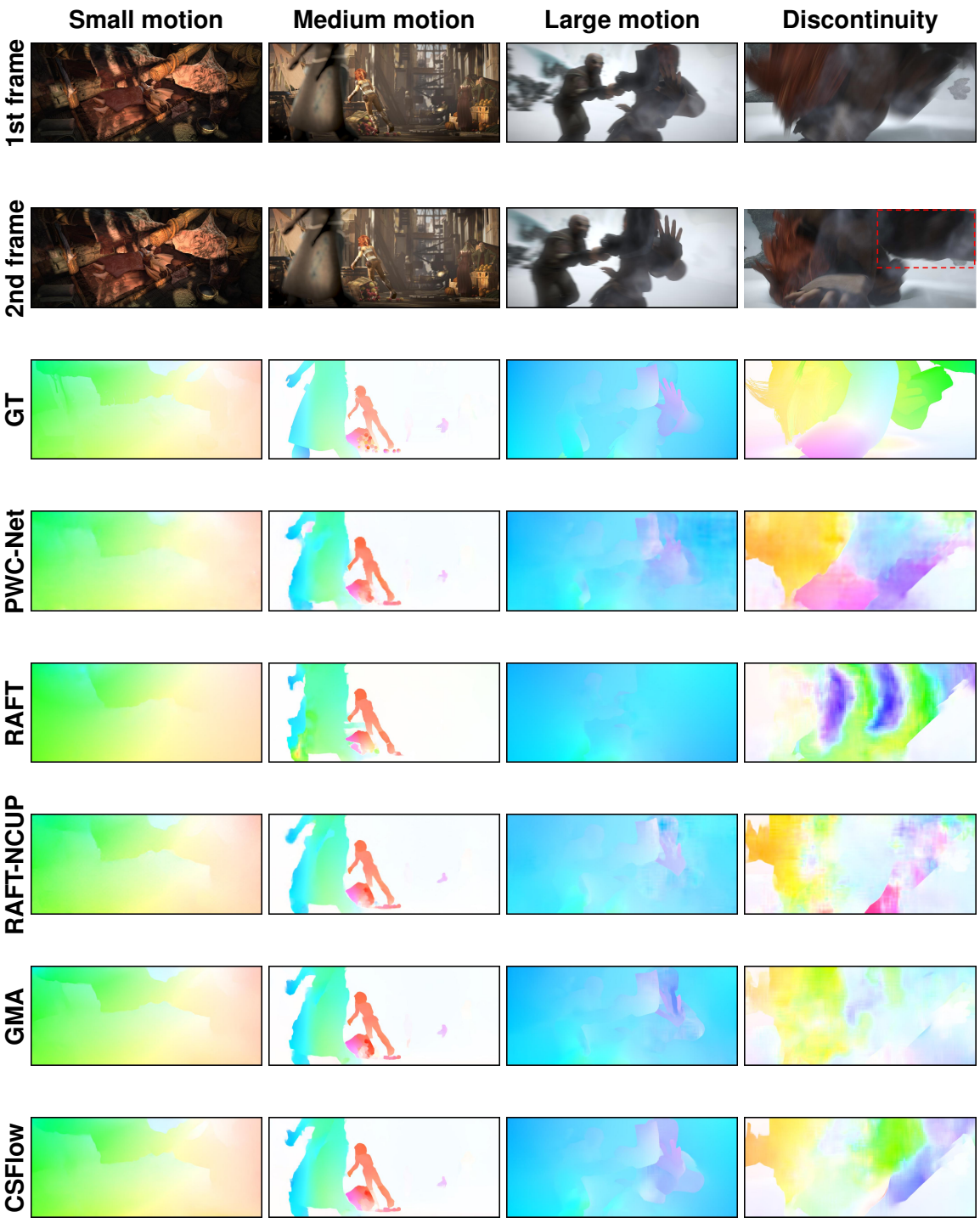


Figure 8.4. *Color-coded results of optical flow estimated by deep learning-based methods for four different scenarios: small, medium, large motion, and discontinuity.*

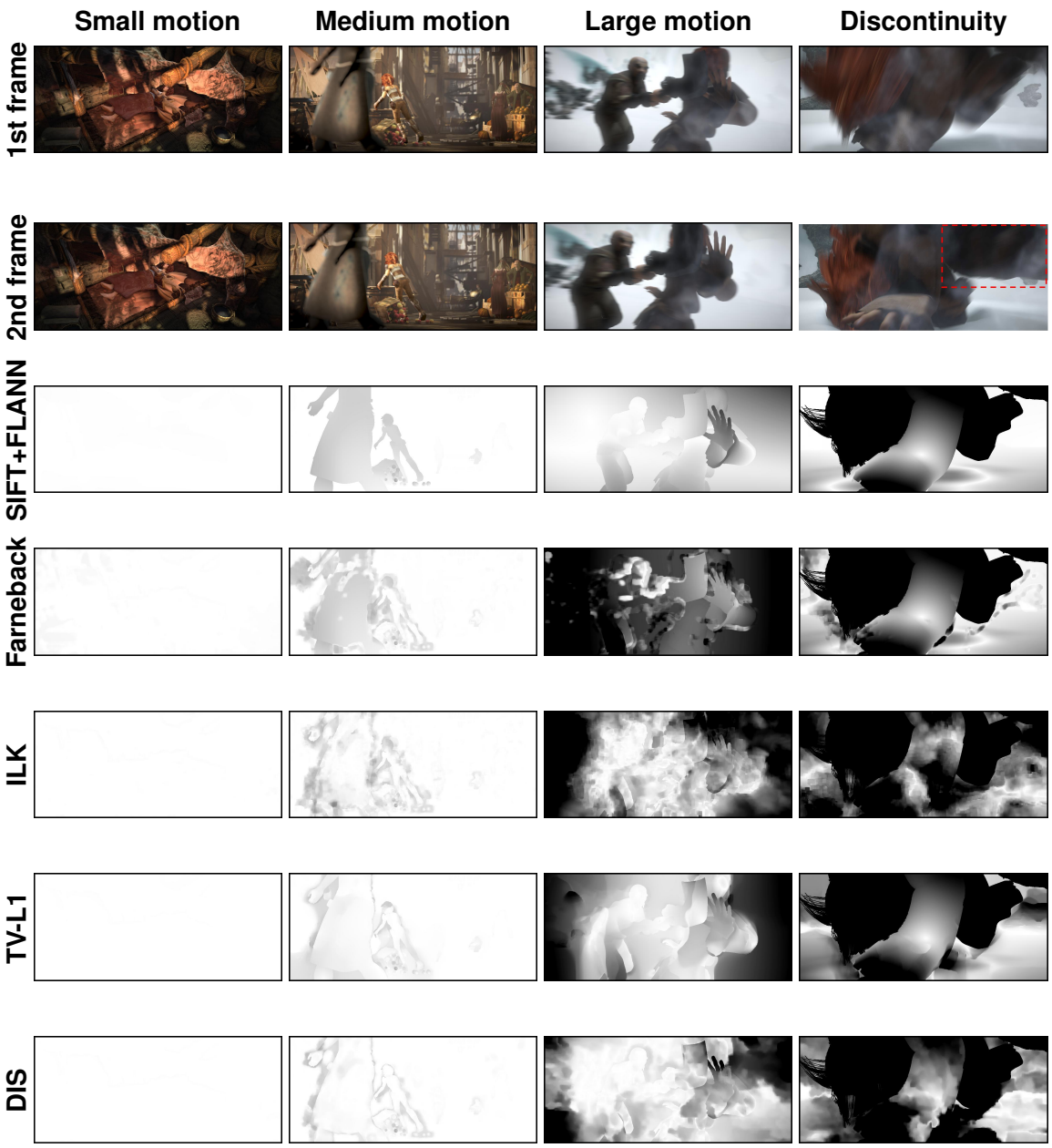


Figure 8.5. *The results of the grayscale absolute error of optical flow estimated by conventional methods for four different scenarios: small, medium, large motion, and discontinuity. White regions indicate where the motion is correctly estimated, while black regions refer to the motion is not correctly estimated. The grayscale images range between [0, 1].*

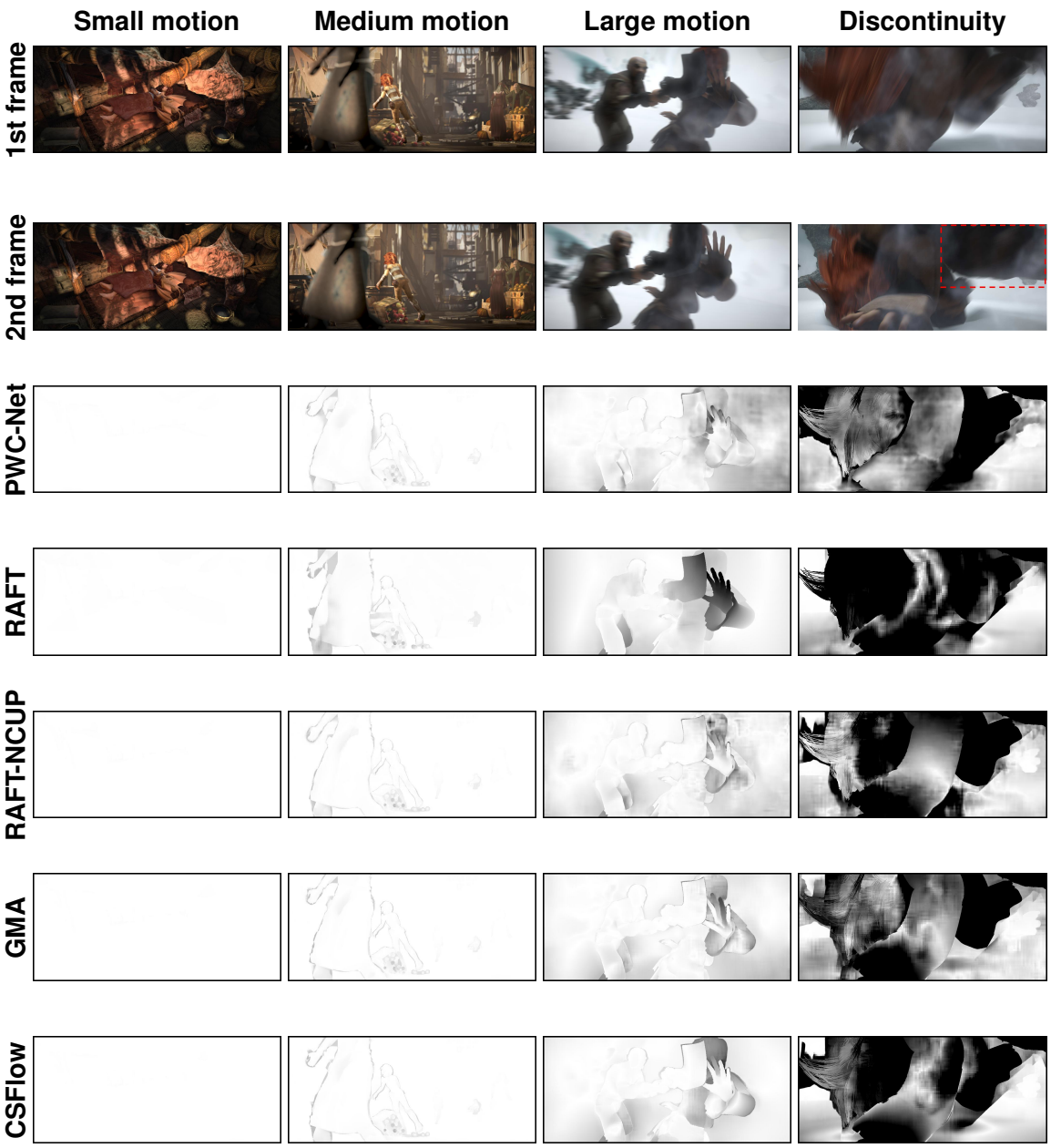


Figure 8.6. *The results of the grayscale absolute error of optical flow estimated by deep learning-based methods for four different scenarios: small, medium, large motion, and discontinuity. White regions indicate where the motion is correctly estimated, while black regions refer to the motion is not correctly estimated. The grayscale images range between [0, 1].*

Table 8.3. *Quantitative results of the AEPE, AEPE in different speed categories, and percentage of error for conventional methods. The values are obtained by averaging all the scenes.*

Method	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+
SIFT + FLANN	20.59	8.51	18.04	34.89	46.31	24.13	29.55
Farneback	11.35	2.71	10.27	27.50	50.81	17.95	31.23
ILK	10.41	5.03	10.11	22.94	61.63	14.25	24.12
TV-L1	8.12	2.76	6.72	22.44	63.52	15.07	21.41
DIS	5.82	2.93	6.18	16.51	73.09	13.01	13.90

Table 8.4. *Quantitative results of the AEPE, AEPE in different speed categories, and percentage of error for deep learning-based methods. The values are obtained by averaging all the scenes.*

Method	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+
PWC-Net	3.70	2.17	3.95	10.68	79.72	11.23	9.04
RAFT	2.68	1.54	3.16	8.94	85.28	8.53	6.19
RAFT-NCUP	3.03	1.80	3.59	9.41	82.79	9.89	7.32
GMA	2.74	1.48	3.10	9.18	85.15	8.56	6.29
CSFlow	2.61	1.50	2.91	8.65	85.34	8.62	6.05

8.1.3 Results on the Flying Chairs Dataset

No experiment has been conducted on deep learning-based methods for the Flying Chairs dataset, as the Flying Chairs dataset is used in pre-trained models. The results are obtained only for conventional methods.

Image pairs, ground truth, and color-coded results of estimated optical flow for four image pairs are depicted in Fig. 8.7. From left-most to right-most, columns show the images for relatively small motion under daylight conditions, large motions, relatively small motion under low light conditions, and scene with occlusion, where the region indicated by the red rectangle. Image pairs, ground truth, and grayscale absolute error results are illustrated in Fig. 8.8. Moreover, quantitative results of the AEPE, AEPE in different speed categories, and percentage of error for the entire dataset are provided in Table 8.5.

Figure 8.7. *Color-coded results of optical flow estimated by conventional methods for four different scenarios: small motion under daylight conditions, large motion, small motion under low light conditions, and occlusion.*

Figure 8.8. *The results of the grayscale absolute error of optical flow estimated by conventional methods for four different scenarios: small motion under daylight conditions, large motion, small motion under low light conditions, and occlusion. White regions indicate where the motion is correctly estimated, while black regions refer to the motion is not correctly estimated. The grayscale images range between [0, 1].*

Table 8.5. *Quantitative results of the AEPE, AEPE in different speed categories, and percentage of error.*

Method	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+
SIFT + FLANN	7.92	4.54	15.70	45.56	64.35	12.91	22.73
Farneback	8.37	3.60	12.97	46.19	50.18	16.40	33.41
ILK	23.47	8.60	44.32	48.03	57.08	11.39	32.52
TV-L1	5.09	2.75	8.74	40.68	63.38	16.96	19.65
DIS	4.67	3.06	7.12	37.42	70.50	13.45	16.04

8.1.4 Results on the In-House Dataset

Section 8.1.4.a presents the visual results and quantitative results for different pre-processing steps, while Section 8.1.4.b presents the visual results and quantitative results for different levels of noise.

8.1.4.a Result on the In-House Dataset for Different Pre-processing

The performance of the optical flow methods is tested on the in-house dataset for different pre-processing steps: raw images, raw images with Black Level Correction (BLC) and Lens Shading Correction (LSC), raw images with BLC, LSC, and White Balance (WB), and raw images with BLC, LSC, WB, and Opto-Electronic Transfer Function (OETF). Image pair, ground truth, and color-coded results of optical flow estimation are illustrated in Fig. 8.9 and 8.10 for conventional and deep learning-based methods, respectively. From left-most to right-most, columns correspond to images for raw images, raw images with BLC and LSC, raw images with BLC, LSC, and WB, and raw images with BLC, LSC, WB, and OETF. Grayscale absolute error results of estimated optical flow by conventional methods and deep learning-based methods are shown in Fig. 8.11 and 8.12, respectively. Furthermore, quantitative results are presented in Table 8.6 and 8.7 for conventional and deep learning-based methods, respectively.

Figure 8.9. *Color-coded results of optical flow estimated by conventional methods for different pre-processing steps.*

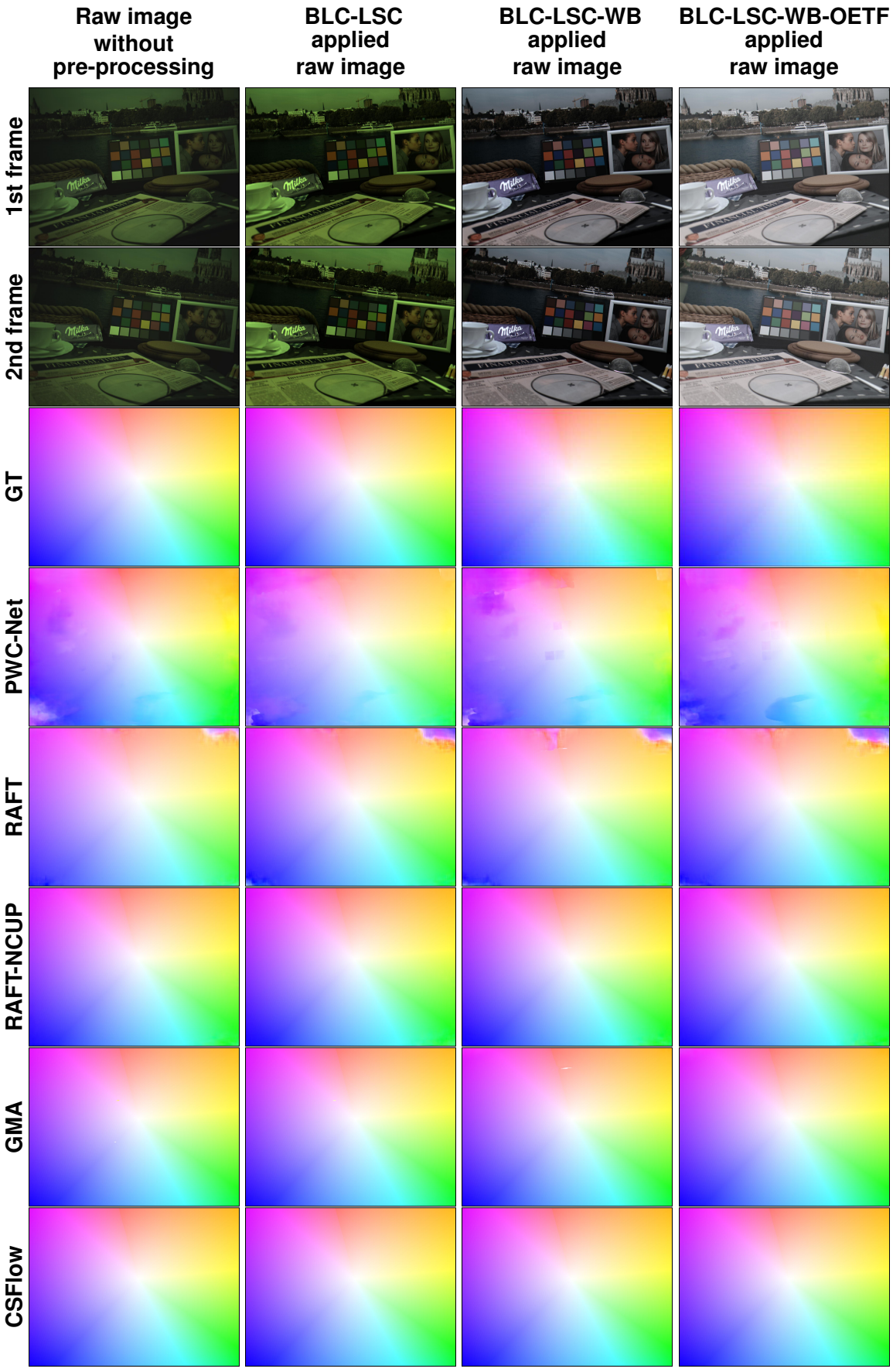


Figure 8.10. *Color-coded results of optical flow estimated by deep learning-based methods for different pre-processing steps.*

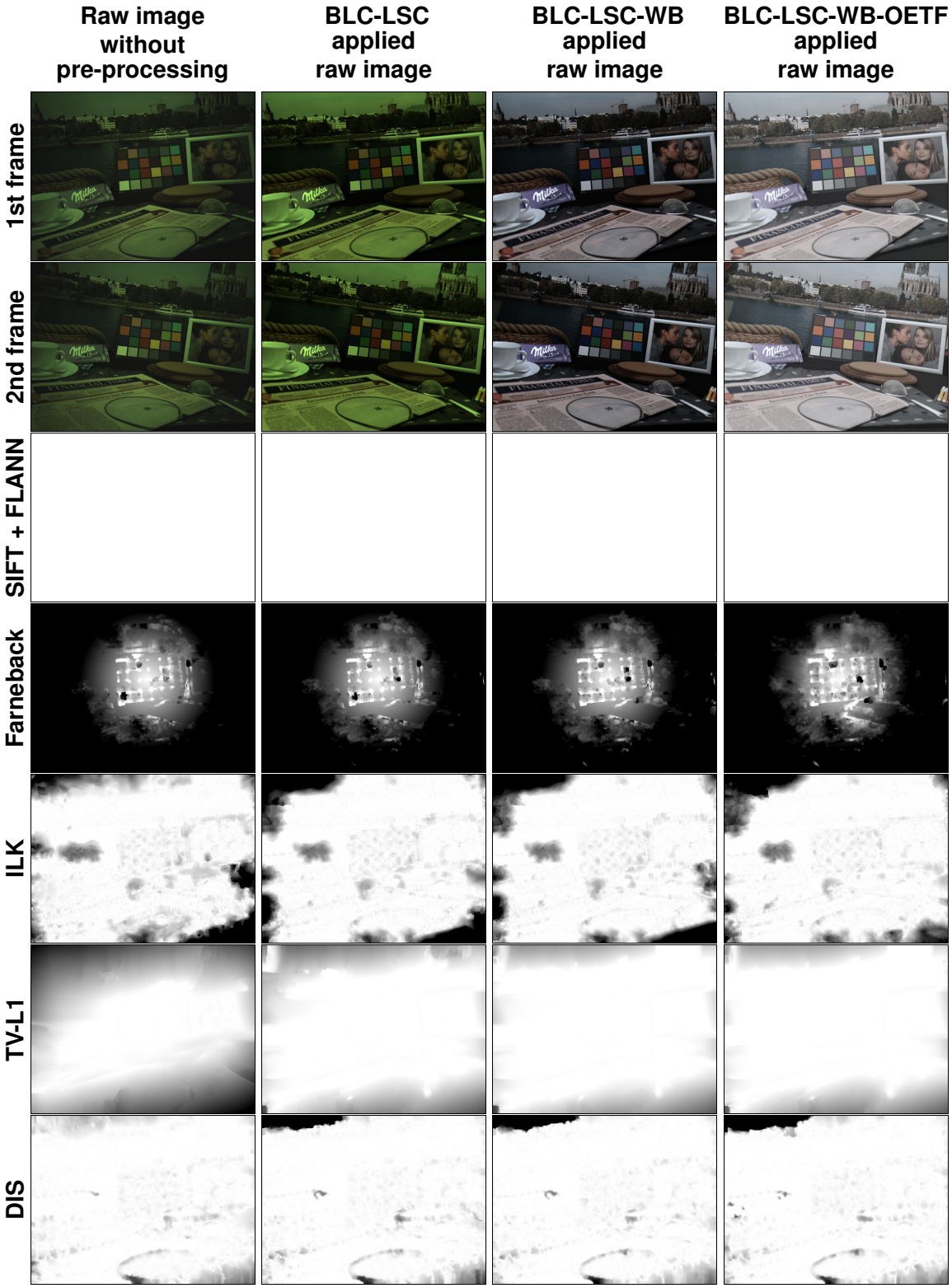


Figure 8.11. *Grayscale absolute error results of optical flow estimated by conventional methods for different pre-processing steps. White regions indicate where the motion is correctly estimated, while black regions refer the motion is not correctly estimated. The grayscale images range between [0, 1].*

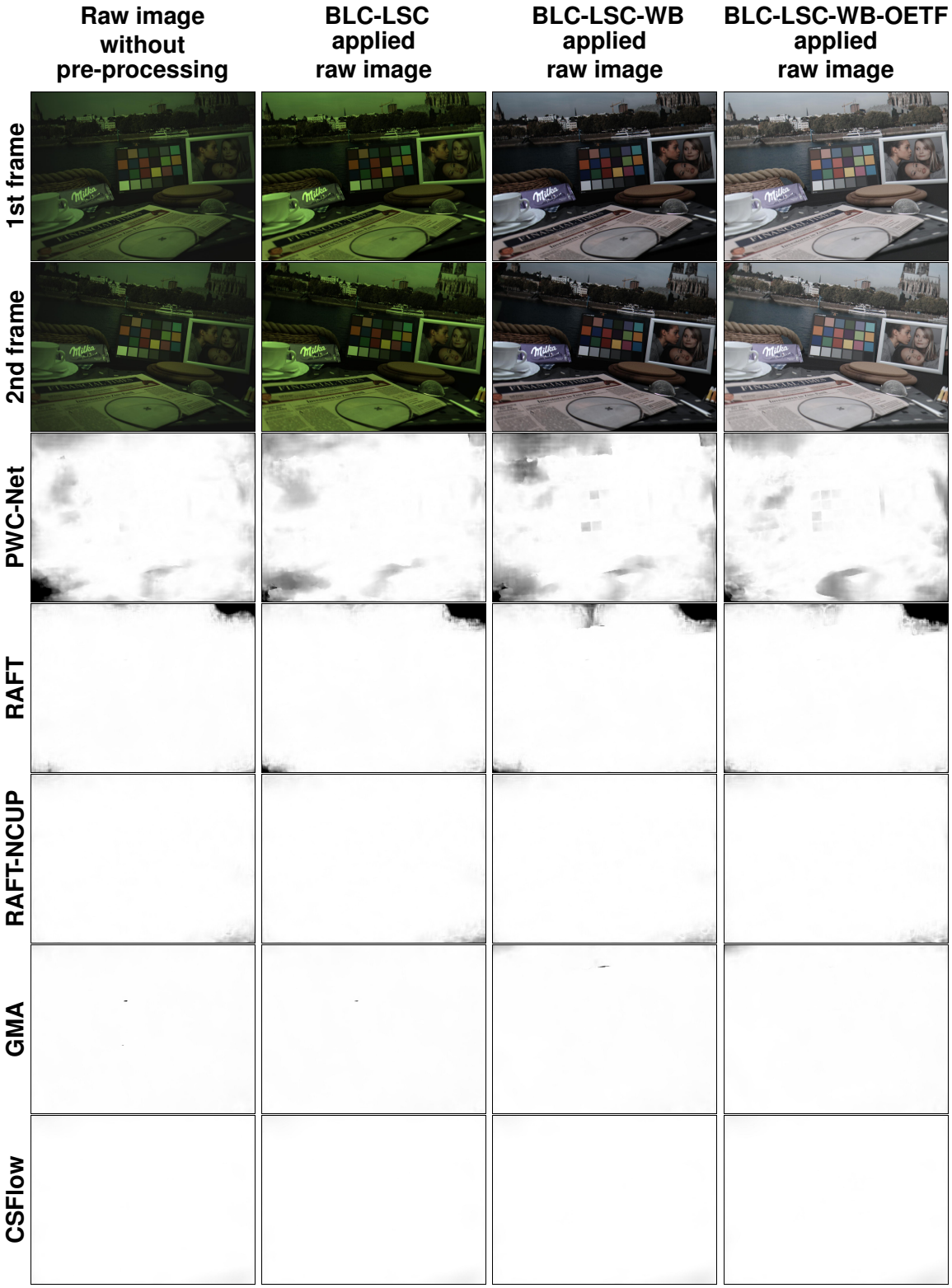


Figure 8.12. *Grayscale absolute error results of optical flow estimated by deep learning-based methods for different pre-processing steps. White regions indicate where the motion is correctly estimated, while black regions refer the motion is not correctly estimated. The grayscale images range between [0, 1].*

Table 8.6. *Quantitative results of the AEPE, AEPE in different speed categories, and percentage of error for different pre-processing step on the in-house dataset for conventional methods.*

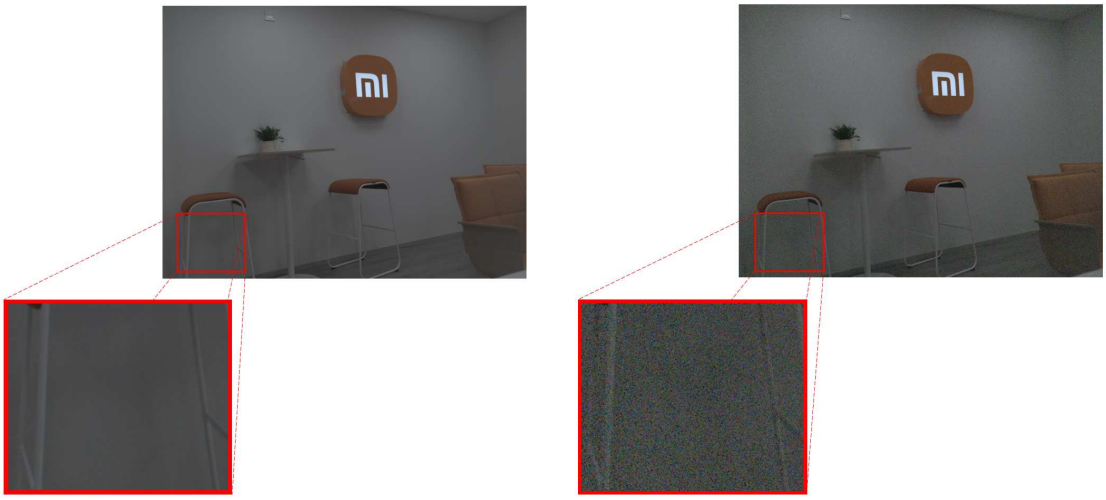
Scenario	Method	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+
Raw images without pre-processing	SIFT + FLANN	17.09	2.39	6.00	18.25	67.23	9.17	23.54
	Farneback	83.85	3.66	22.54	91.94	1.46	0.95	97.59
	ILK	30.27	1.71	6.42	33.74	31.67	18.93	49.39
	TV-L1	40.64	9.37	13.57	44.14	14.01	10.18	75.80
	DIS	9.06	1.58	2.96	9.92	56.40	19.85	23.75
BLC-LSC applied raw images	SIFT + FLANN	5.48	0.78	1.64	5.80	83.45	7.85	8.70
	Farneback	83.37	3.10	20.71	91.75	2.51	1.29	96.19
	ILK	19.27	3.71	5.17	21.34	62.17	13.36	24.47
	TV-L1	22.26	7.87	9.42	23.85	39.04	11.38	49.58
	DIS	5.12	1.68	2.26	5.55	66.03	18.99	14.99
BLC-LSC-WB applied raw images	SIFT + FLANN	23.82	3.29	16.18	24.52	87.11	6.23	6.67
	Farneback	83.12	2.85	19.73	91.65	3.16	1.45	95.40
	ILK	18.80	3.31	5.41	20.79	63.51	13.63	22.86
	TV-L1	18.23	6.54	7.82	19.55	45.36	11.62	43.02
	DIS	4.99	1.48	2.05	5.41	67.10	18.41	14.49
BLC-LSC-WB-OETF applied raw images	SIFT + FLANN	1.36	0.24	0.42	1.42	93.75	4.20	2.06
	Farneback	82.34	2.19	16.91	91.25	4.57	1.90	93.53
	ILK	17.17	2.82	4.94	19.02	66.16	14.14	19.70
	TV-L1	12.82	4.02	4.84	13.82	55.03	11.33	33.64
	DIS	4.72	1.44	1.88	5.14	68.74	17.35	13.90

Table 8.7. *Quantitative results of the AEPE, AEPE in different speed categories, and percentage of error for different pre-processing step on the in-house dataset for deep learning-based methods.*

Scenario	Method	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+
Raw images without pre-processing	PWC-Net	22.03	3.38	5.28	23.60	45.22	16.36	38.41
	RAFT	43.83	9.77	12.74	47.56	60.34	12.78	26.89
	RAFT NCUP	9.67	1.62	2.58	10.53	75.32	10.87	13.80
	GMA	12.66	4.79	6.15	13.22	53.74	34.77	11.49
	CSFlow	3.74	1.32	2.07	3.94	83.21	9.15	7.64
BLC-LSC applied raw images	PWC-Net	21.27	5.08	7.24	22.60	46.36	17.70	35.93
	RAFT	41.08	9.52	12.73	44.59	63.28	11.26	25.46
	RAFT NCUP	7.39	2.12	2.52	8.02	77.96	10.63	11.41
	GMA	10.41	3.66	5.07	10.97	59.32	29.84	10.84
	CSFlow	2.33	1.03	1.53	2.42	85.86	8.68	5.46
BLC-LSC-WB applied raw images	PWC-Net	20, 35	3, 73	5, 90	21, 73	48, 49	16, 13	35, 37
	RAFT	44.29	11.63	15.41	47.88	62.06	10.82	27.12
	RAFT NCUP	8.19	2.46	3.33	8.86	77.02	10.84	12.14
	GMA	10.59	4.01	5.75	11.14	64.32	24.14	11.54
	CSFlow	2.84	1.64	1.84	2.91	85.59	8.72	5.69
BLC-LSC-WB-OETF applied raw images	PWC-Net	21.99	5.36	7.63	23.36	45.04	18.60	36.35
	RAFT	39.88	9.73	12.87	43.17	63.11	11.06	25.83
	RAFT NCUP	7.27	2.26	2.83	7.83	78.20	10.60	11.19
	GMA	9.22	3.62	4.66	9.73	69.83	19.16	11.01
	CSFlow	2.40	1.78	1.92	2.34	86.35	8.59	5.06

8.1.4.b Result on the In-House Dataset for Different Levels of Noise

The performance of optical flow methods is evaluated on the processed (BLC-LSC, WB, OETF applied) in-house dataset for different levels of noise. Pre-calculated noise parameters at five different Analog Gain (AG) (AG 1, AG 4, AG 16, AG 32, AG 64) are used to generate noise. Fig. 8.13 shows an image from the in-house dataset where the AG is 1 and 64. Image pair, ground truth, and color-coded results are illustrated in Fig. 8.14 and 8.15 for conventional methods and deep learning-based methods, respectively. Grayscale absolute error results of conventional methods and deep learning-based methods are shown in Fig. 8.16 and 8.17, respectively. Furthermore, quantitative results are presented in Table 8.8 and 8.9 for conventional and deep learning-based methods, respectively.



(a) AG 1 ***(b)*** AG 64

Figure 8.13. *Same image at different levels of noise.*

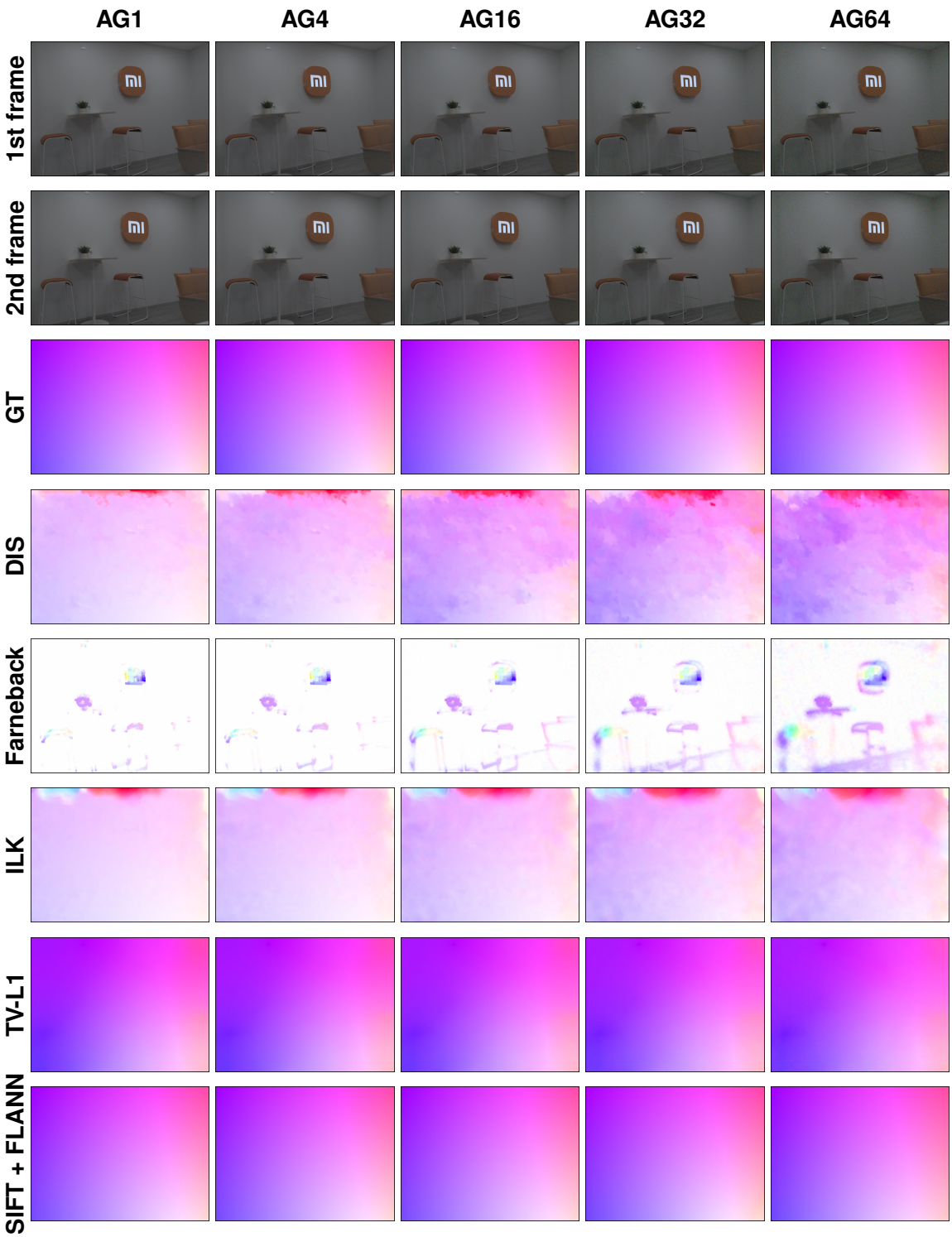


Figure 8.14. *Color-coded results of optical flow estimated by conventional methods for five different levels of noise.*

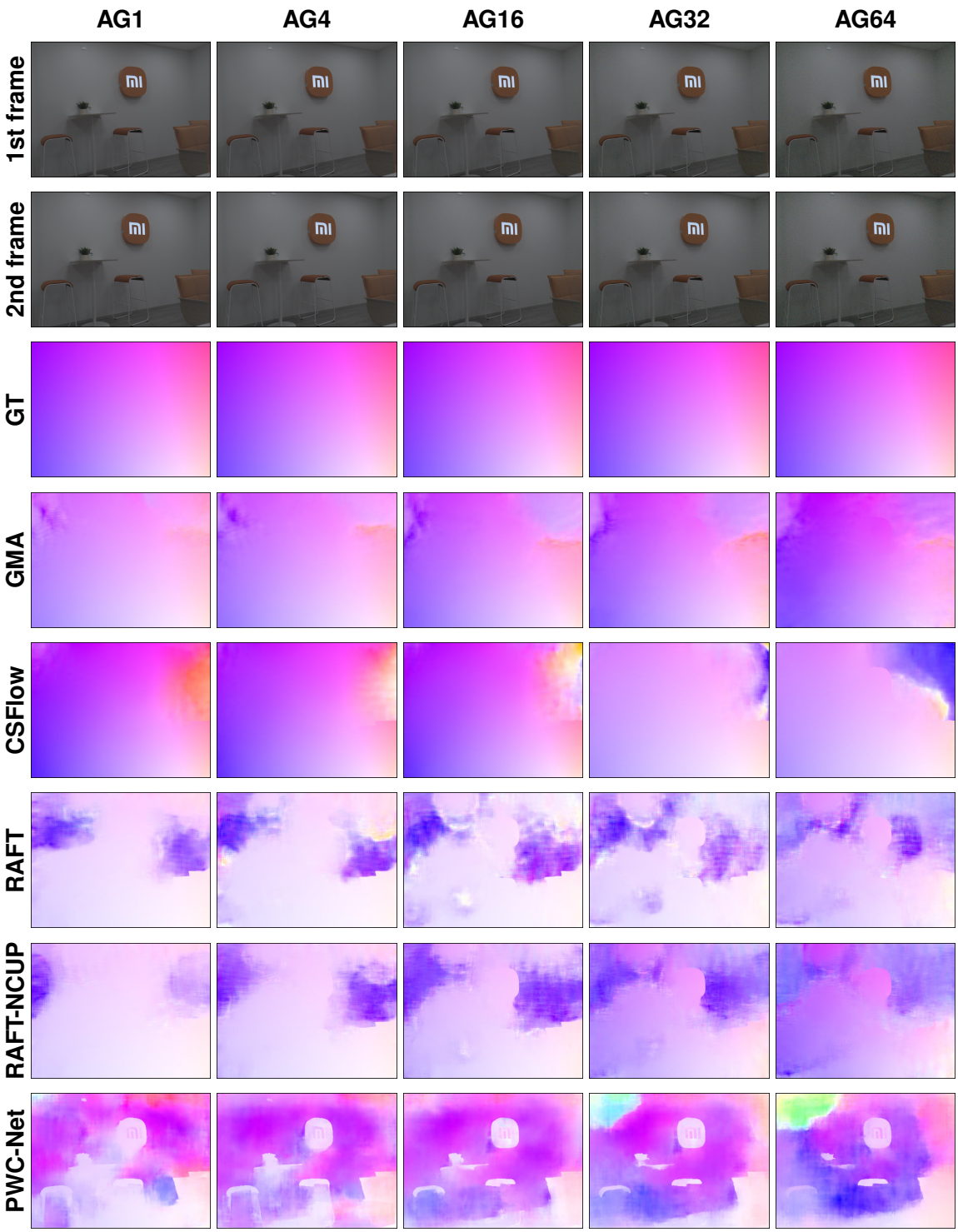


Figure 8.15. *Color-coded results of optical flow estimated by deep learning-based methods for five different levels of noise.*

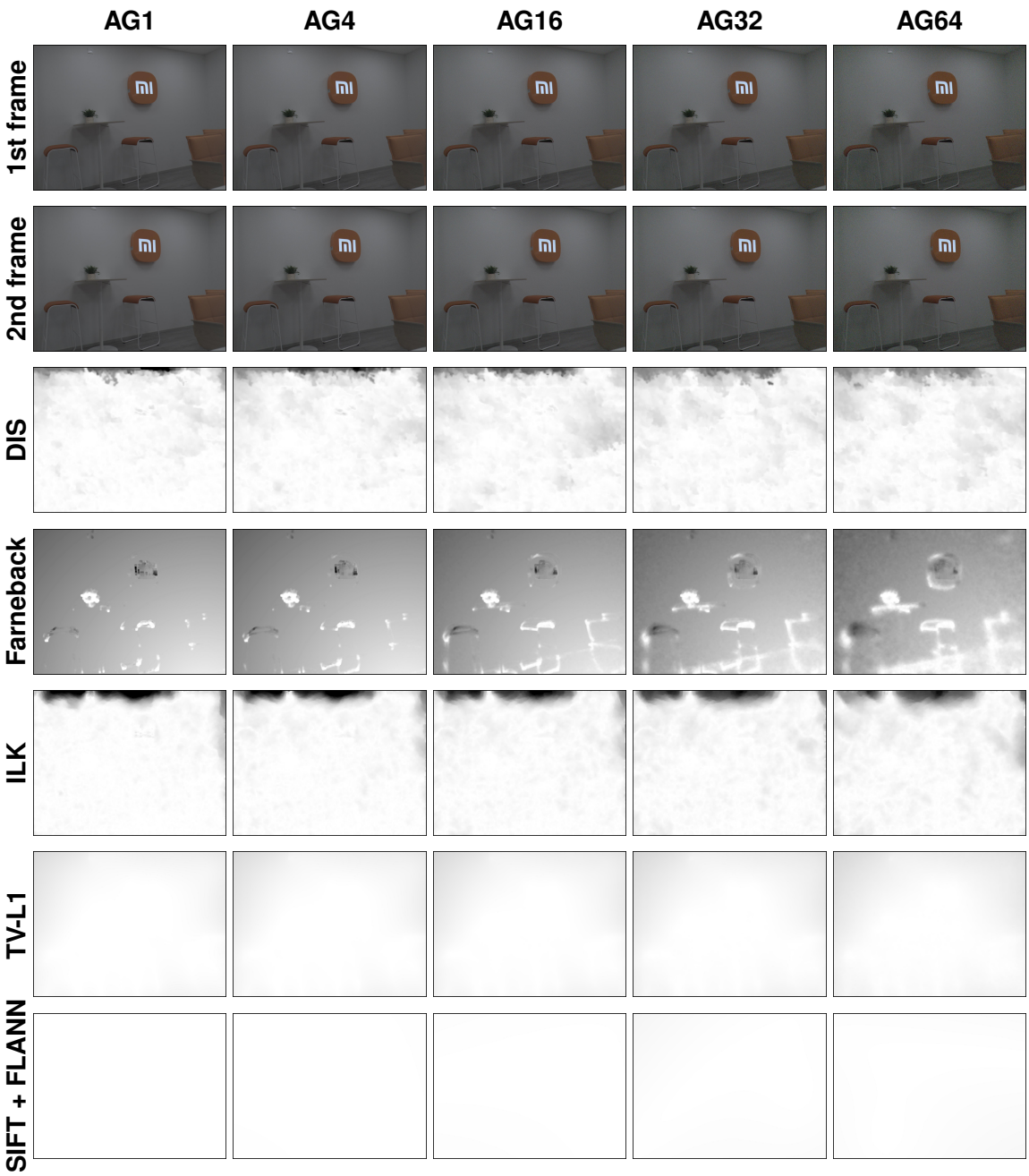


Figure 8.16. *Grayscale absolute error results of optical flow estimated by conventional methods for five different levels of noise. White regions indicate where the motion is correctly estimated, while black regions refer the motion is not correctly estimated. The grayscale images range between [0, 1]*

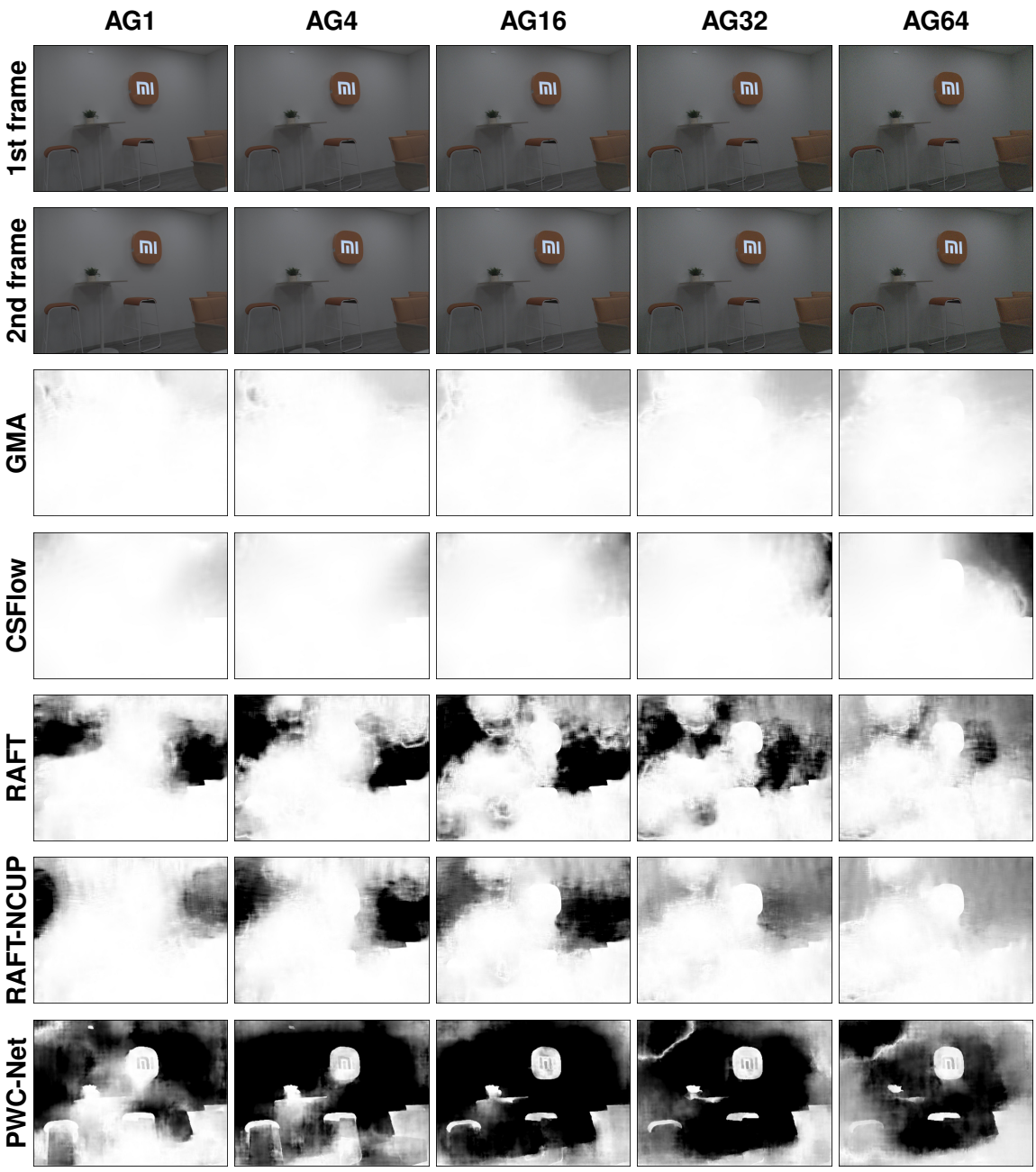


Figure 8.17. *Grayscale absolute error results of optical flow estimated by deep learning-based methods for five different levels of noise.White regions indicate where the motion is correctly estimated, while black regions refer the motion is not correctly estimated. The grayscale images range between [0, 1].*

Table 8.8. *Quantitative results of the AEPE, AEPE in different speed categories, and percentage of error at different AG for conventional methods.*

AG (noise level)	Method	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+
AG 1	SIFT + FLANN	1.78	0.20	0.57	1.94	93.41	4.44	2.14
	Farneback	81.33	1.97	17.10	90.08	5.04	1.84	93.11
	ILK	17.94	4.03	6.12	19.49	68.04	12.09	19.85
	TV-L1	12.66	3.46	4.71	13.47	56.85	10.74	32.40
	DIS	5.31	1.76	2.38	5.67	71.27	15.38	13.34
AG 4	SIFT + FLANN	1.72	0.21	0.54	1.89	93.56	4.14	2.29
	Farneback	81.19	1.92	16.69	89.98	5.10	1.96	92.93
	ILK	17.46	3.74	6.35	18.85	65.26	14.89	19.83
	TV-L1	12.71	3.50	4.76	13.52	55.96	11.38	32.64
	DIS	5.56	1.63	2.37	5.96	67.23	17.81	14.95
AG 16	SIFT + FLANN	2.18	0.40	0.77	2.38	91.20	5.91	2.87
	Farneback	80.74	1.77	15.45	89.67	5.31	2.39	92.28
	ILK	14.99	3.03	5.02	16.30	60.31	19.70	19.98
	TV-L1	12.76	3.57	4.84	13.58	54.52	12.39	33.08
	DIS	5.77	1.78	2.70	6.11	60.21	22.71	17.07
AG 32	SIFT + FLANN	2.31	0.56	0.87	2.46	89.36	8.09	2.53
	Farneback	80.27	1.62	14.29	89.33	5.58	2.92	91.49
	ILK	14.04	3.16	5.15	15.25	56.40	23.30	20.28
	TV-L1	12.81	3.62	4.91	13.62	53.40	13.15	33.43
	DIS	5.80	2.11	3.00	6.10	55.02	26.48	18.48
AG 64	SIFT + FLANN	7.05	5.38	4.99	7.51	83.07	12.27	4.52
	Farneback	79.73	1.49	13.08	88.93	5.97	3.17	90.30
	ILK	14.21	3.17	4.96	15.48	51.17	27.62	21.19
	TV-L1	12.92	3.73	5.02	13.74	51.62	14.30	34.06
	DIS	6.59	2.45	3.35	6.94	48.32	31.24	20.42

Table 8.9. *Quantitative results of the AEPE, AEPE in different speed categories, and percentage of error at different AG for deep learning-based methods.*

AG (noise level)	Method	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+
AG 1	PWC-Net	19.54	5.09	5.94	20.53	50.54	17.01	32.44
	RAFT	9.98	4.32	4.61	10.43	78.95	9.75	11.28
	RAFT-NCUP	5.95	1.25	1.38	6.43	80.58	9.70	9.70
	GMA	2.22	1.38	1.39	2.30	87.67	7.23	5.09
	CSFlow	1.84	1.06	1.24	1.95	88.38	7.53	4.08
AG 4	PWC-Net	24.84	7.79	10.07	25.74	40.89	19.84	39.21
	RAFT	14.68	7.62	8.43	14.97	73.31	11.79	14.89
	RAFT-NCUP	9.01	3.83	4.05	9.36	76.30	11.10	12.58
	GMA	3.37	1.92	2.23	3.48	84.29	8.75	6.94
	CSFlow	2.45	1.23	1.63	2.58	84.67	9.38	5.39
AG 16	PWC-Net	30.57	10.28	13.49	31.77	28.42	22.30	40.26
	RAFT	22.81	9.65	12.25	23.49	62.72	15.13	22.13
	RAFT-NCUP	14.91	6.69	7.35	15.38	68.04	13.50	18.44
	GMA	5.35	2.87	3.11	5.56	78.80	11.86	9.33
	CSFlow	3.90	3.08	2.95	3.88	78.80	12.03	9.16
AG 32	PWC-Net	33.32	11.12	14.74	34.84	21.32	21.64	57.02
	RAFT	25.56	10.13	13.51	26.43	55.37	17.63	26.98
	RAFT-NCUP	15.97	5.62	6.75	16.70	61.94	15.56	22.49
	GMA	6.58	3.37	3.60	6.89	73.99	14.13	11.87
	CSFlow	5.12	2.91	4.20	5.16	74.32	14.59	11.08
AG 64	PWC-Net	37.90	11.22	16.42	39.85	13.89	18.64	67.45
	RAFT	25.26	9.10	12.03	26.35	47.14	20.51	32.33
	RAFT-NCUP	16.90	3.75	5.19	18.02	54.12	17.92	27.94
	GMA	8.40	3.45	3.96	8.91	65.93	16.88	17.17
	CSFlow	5.36	2.82	3.59	5.54	68.35	18.83	12.81

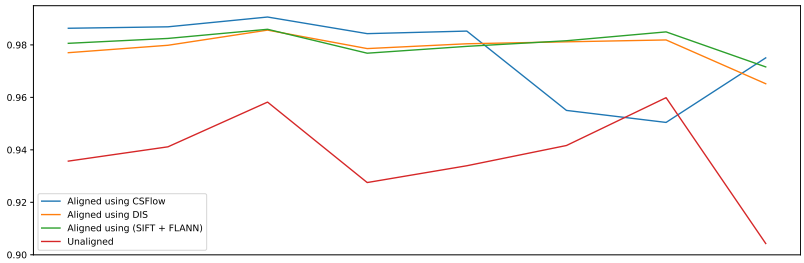
8.2 Results With Fused Images

The quantitative results of the conventional and deep learning-based optical flow estimation are reported in Sections 8.1. Among the motion estimation methods, the three best methods, CSFlow, DIS, and SIFT + FLANN, are selected to estimate pixel shifts between consecutive raw images to align them in raw Bayer domain. Processed (BLC-LSC, WB, OETF applied) raw images and delta images (difference between the reference and target images) before and after alignment are illustrated in Fig. 8.18. Images provided in this section are taken by Xiaomi MI 11 pro smartphone [88].

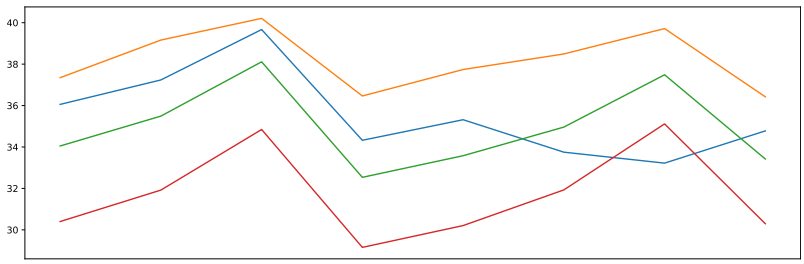
The first column represents the target frames and the reference frame (4th frame). Delta images before alignment are represented in the second column. The other columns, from left to right, represent delta images after aligning the reference and target frames, where pixel shifts are estimated by the CSFlow, DIS, and SIFT + FLANN methods. Five quality assessment methods are used to evaluate the quality of the fused image: Structural Similarity Index Measurement (SSIM), Peak Signal to Noise Ratio (PSNR), Spectral Angle Mapper (SAM), Universal Quality Image Index (UQI), and Correlation Coefficient (CC). The results are presented in Fig. 8.19. The red line represents the quality assessment between the reference and unaligned target frames (i.e., RF - F1 refers to the quality

assessment between the reference frame (RF) and first frame (F1), $\mathtt{all}$ refers to the fusion of all frames). Blue, green, and orange lines represent the quality assessment between the RF and aligned target frames where the pixel shifts are estimated using CSFlow, (SIFT + FLANN), and DIS methods, respectively. Lastly, fusion of multi-frames before and after alignment is depicted in Fig. 8.20.

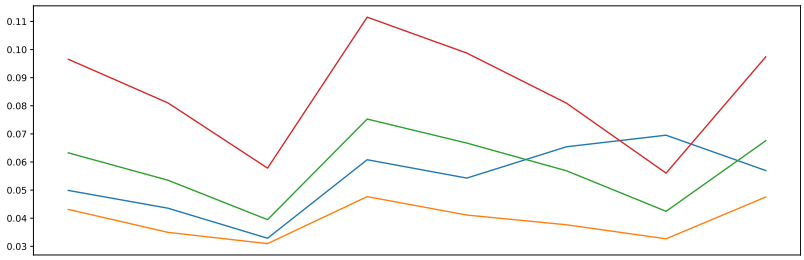
Figure 8.18. *Processed raw image frames and delta images before and after alignment. From left-most to right-most, columns refer to multi frames, delta images before alignment, delta images after alignment, where pixel shifts are estimated by the CSFlow, DIS, and SIFT + FLANN methods, respectively.*



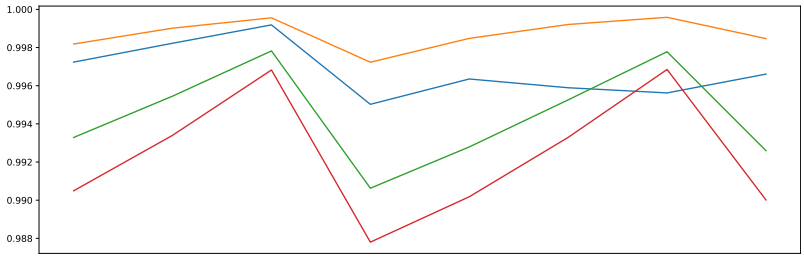
(a) SSIM scores



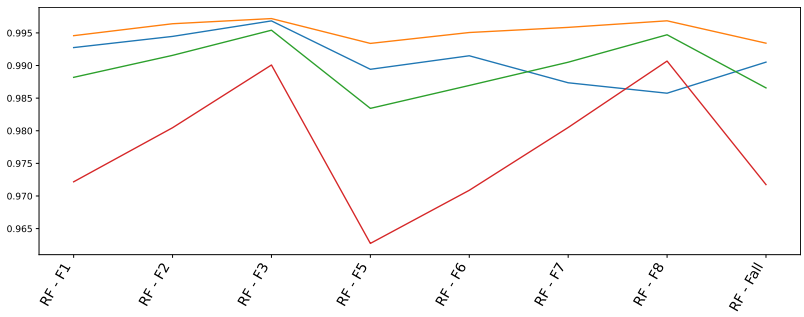
(b) PSNR scores



(c) SAM scores



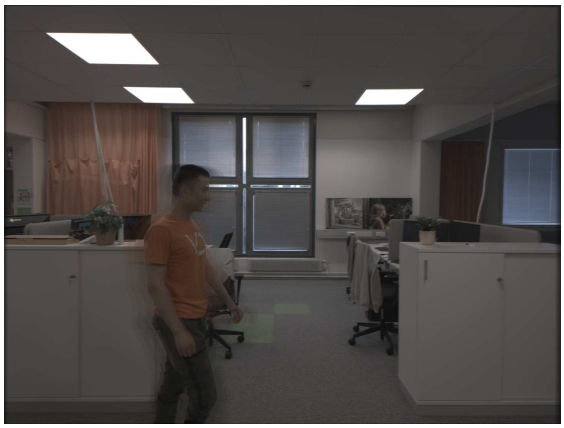
(d) UQI scores



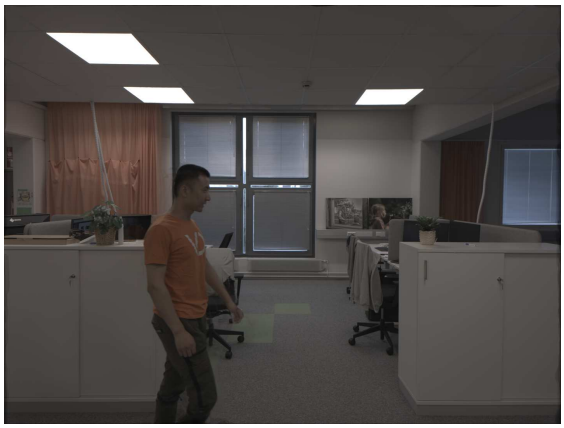
(e) CC scores

Figure 8.19. *Plot results of image fusion for different quality assessment method.*

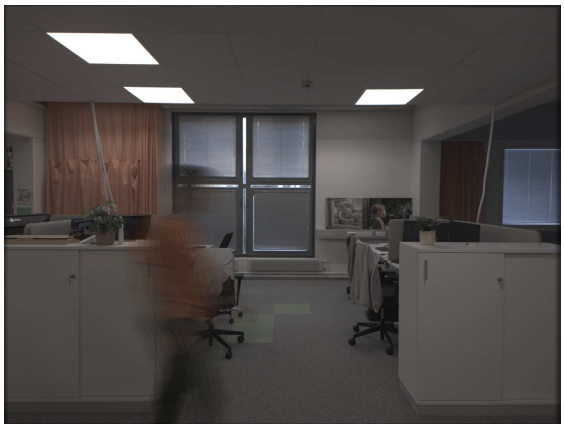
(a) Fusion of processed eight raw frames before alignment



(b) Fusion after alignment where the pixel shifts between the reference and each target frame are estimated by CSFlow method



(c) Fusion after alignment where the pixel shifts between the reference and each target frame are estimated by DIS method



(d) Fusion after alignment where the pixel shifts between the reference and each target frame are estimated by SIFT + FLANN method

Figure 8.20. *Fusion of eight processed raw frames before and after alignment*

8.3 Results With Fused Images for Different Demosaicing Algorithms

In this subsection, the effect of two different demosaicing algorithms on the quality of image alignment is studied. The first method is bilinear interpolation, where the missing color channels are interpolated, and the resulting image has the same resolution as the Bayer image. The second method is one of the two green channels skipping mode, where one of the two green channels is discarded, and this causes image resolution to drop in half. It does not correct the 1 pixel shift in the spatial location of each color component. In the 2x2 Bayer quad, the R, G, G, and B pixels correspond to different spatial locations on the camera sensor. If this 2x2 pixel Bayer quad is treated as RGB pixel by dropping one of the green channels, then this 1 pixel phase difference remains on top of halving the resolution. The processed raw images used for this test are depicted in Fig. 8.21. Delta images are illustrated in Fig. 8.22. All target frames are aligned using DIS and

CSFlow methods. Then, the aligned target frame and the reference frame are fused. Finally, a delta image is created by taking the difference between the fused image and the reference image. As mentioned, the image resolution drops in half when omitting one of the two green channels. As a result, estimated optical flow also has half resolution of the Bayer image. Therefore, estimated optical flow is interpolated to Bayer image resolution for comparison. Five quality assessment methods are used to evaluate the quality of the fused image, and the results are presented in Table 8.10.

Figure 8.21. *Multi frames.*

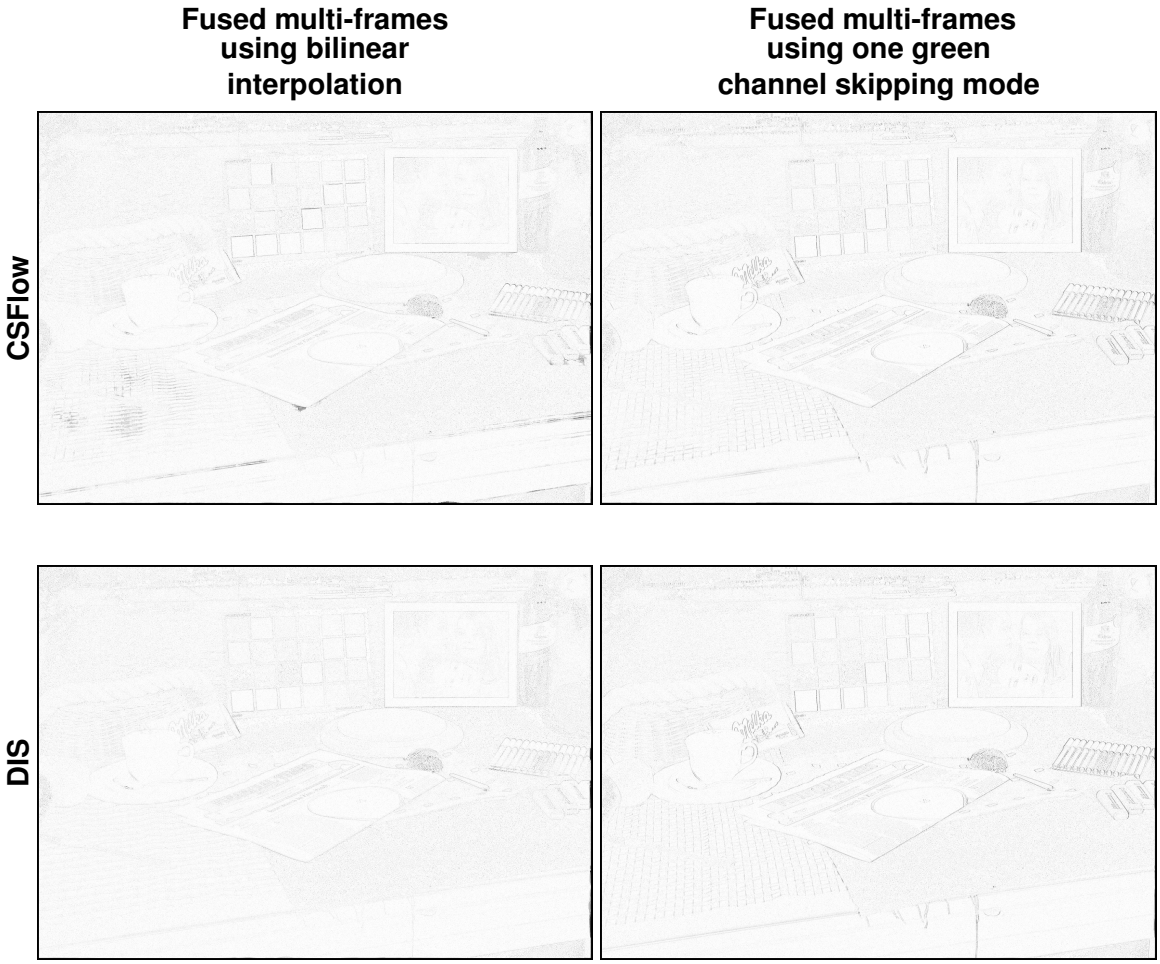


Figure 8.22. *Delta images of aligned multi-frame fusion.*

Table 8.10. *Quantitative results of image fusion of two different demosaicing algorithms for different quality assessment method.*

Method	Bilinear interpolation		skipping mode	
	DIS	CSFlow	DIS	CSFlow
SSIM	0.95	0.94	0.94	0.93
PSNR	31.91	31.53	31.66	31.46
SAM	0.06	0.06	0.07	0.07
UQI	0.98	0.98	0.96	0.95
CC	0.99	0.99	0.99	0.99

8.4 Discussion of the Results

Discussion on the Results of the Sintel Dataset

The final pass results of optical flow estimation are shown in Fig. 8.1 for conventional methods and Fig. 8.2 for deep-learning-based methods. It is clear that the performance of optical flow methods showed dependency on the amount of existing motion in the scene. The larger motion in the scene causes higher errors, while the smaller motion in the scene causes lower errors, as seen in Table 8.2. Furthermore, the results show that the AEPEs of the deep learning-based methods have lower magnitude than conventional methods. Which means that deep learning-based methods are more robust than conventional methods in optical flow estimation.

As can be seen from Table 8.3 and Table 8.4, in general, deep learning-based methods perform better than conventional methods. CSFlow method outperforms other methods. SIFT + FLANN method produces the highest error among all methods. This is expected because SIFT + FLANN algorithm can only predict global camera motion and fails in object motion estimation. Among the conventional methods, the DIS method achieves the best results as it aims to keep the execution time low and the accuracy high. Among the deep learning methods, transformer-based methods, GMA and CSFlow, perform better than CNN or CNN-RNN-based methods, PWC-Net, RAFT, and RATF-NCUP, as attention mechanism in transformers is known to improve the performance of various networks in image processing. CNN-based method, PWC-Net, has the highest error among deep-learning methods. Furthermore, It is clear that optical flow methods do not perform as desired in the presence of discontinuity as seen in Fig. 8.3 and 8.4

Discussion on the Results of the Flying Chairs Dataset

From Table 8.5, it can be observed that the DIS method produces the best results, while ILK has the worst results. The performance of SIFT + FLANN is better than the result in the Sintel dataset. The reason for this may be that the objects in the Flying Chairs are small and take up small area in the whole image. In parallel, more area can be well estimated globally by SIFT + FLANN method. Moreover, The findings indicate that low light conditions affect the performance of optical flow methods in negative way, as can be seen from Fig. 8.7 and Fig. 8.8

Discussion on the Results of the In-House Dataset for Different Pre-processing Steps

Quantitative results are provided in the Table 8.6 for conventional methods and Table 8.7 for deep learning-based methods. The results show that optical flow algorithms generally achieve remarkable overall performance after each pre-processing step, and the best results are obtained after applying Black Level Correction (BLC), Lens Shading Correction (LSC), White Balance (WB), and Opto-Electronic Transfer Function (OETF). It is clear that SIFT + FLANN outperforms other methods after applying BLC, LSC, WB, and OETF.

This is expected since the images in the In-House dataset contain only global camera motion, that is, no object motion in the images. CSFlow method achieves the best results, while the RAFT method has the highest error among deep-learning methods after each pre-processing step. The farneback method has the highest error among all methods. As seen from Fig. 8.14 and Fig. 8.15, the farneback method detects some distinctive features of the stationary objects in the scene such as object edges and fails in global camera motion estimation. This may be the reason why the farneback method performs poorly. Deep learning-based algorithms were trained using RBG images. So algorithms may be sensitive to color information. BLC-LSC-WB-OETF applied raw images are closer to images used for training in terms of color information. This may be the reason why deep learning based optical flow algorithms perform better on BLC-LSC-WB-OETF applied raw images.

Discussion on the Results of the In-House Dataset for Different Levels of Noise

Based on the quantitative results given in Table 8.8 and Table 8.9, the noise affects the performance of optical flow algorithms. In parallel with the increase in noise, the motion estimation is more error-prone. SIFT + FLANN achieves the best results among conventional methods, while the farneback method has the highest error in the noisy environment due to the reason stated above. It can be seen that the TV-L1 method is not affected by noise compared to the other algorithms since the TV-L1 method offers more accurate and reliable results in noisy environments. The best motion estimation for all deep learning-based methods is obtained where noise is lowest (AG1), and the worst estimation is mostly obtained where noise is highest (AG64). The best results are obtained by CSFlow, while PWC-Net produces the worst results among deep learning-based algorithms. In the presence of noise, transformer-based methods, CSFlow and GMA, outperform CNN and CNN-RNN-based PWC-Net, RAFT, and RAFT-NCUP methods. As it can be seen from Fig. 8.14 and Fig. 8.15, the PWC-Net method detects the stationary objects as moving objects (i.e., sign). This may be the reason why the PWC-Net method does not perform well. An image captured under low light conditions, e.g., at night, has a Low Dynamic Range (LDR) and the image includes noise that degrades the quality of the image. The dark area of the image can be enhanced partially by increasing the Analog Gain (AG). As the results shown that the performance of optical flow decreases when the AG increases. Therefore, optical Flow algorithms may require a special denoising strategy for accurate estimation in noisy image sequences.

Based on the quantitative results obtained from all datasets, deep learning-based and conventional methods perform well when the motion between consecutive images is relatively small. However, parallel to the increase in motion, the motion estimation is more error-prone and does not perform as desired. For example, according to the results obtained from Tables 8.3, 8.4, 8.5, 8.6, 8.7, 8.8, and 8.9, the AEPE in the region where the motion is smaller than 10 pixels (s0-10) is lower than s10-40, and s10-40 is lower than

s40+.

Discussion on the Results of Image Alignment and Fusion in Raw Bayer Domain

As shown in Fig. 8.19, in most cases, fusing aligned frames gives better results than fusing unaligned frames. Furthermore, reference and fused frames are most similar when the pixel shifts are estimated by the DIS method, while reference and fused frames are least similar when the pixel shifts are estimated by SIFT + FLANN method. The main reason is that while the other two methods can predict motion globally and locally, SIFT + FLANN only predicts motion globally and fails on object motions.

Fig. 8.20 shows the final output. It is clear that the DIS method produces a result very close to the reference frame and has almost no ghosting artifacts. Furthermore, the CS-Flow method causes slight ghosting artifacts on the moving object. On the other hand, the SIFT + FLANN method significantly solves the misalignment problem caused by global camera motion but fails for the object motion due to the reason given above.

Furthermore, two different methods were used to construct RGB image from raw Bayer image. Delta images are presented in Fig. 8.22 and quantitative results given in Table 8.10. Based on the results obtained, aligning multi-frames based on the demosaicing gives better results than aligning multi-frames based on the one green channel skipping mode. Because in demosaicing, missing colors are interpolated for all pixel locations in Bayer image. In skipping mode, the image resolution is halved because one green channel is dropped. As the optical flow is computed for each pixel location, it provides more accurate alignment compared to the skipping mode. As can be seen from figure 2, misalignment of one green channel skipping mode is higher than misalignment of demosaicing.

9. CONCLUSION AND FURTHER WORK

The misalignment problem arises between raw images due to global camera motion (camera shake) and object movement in the scene. Fusing raw images without alignment causes ghosting artifacts in the final image. To deal with this, the optical flow is computed to determine how much each pixel of the reference image is displaced at the target image. So that the related image content can be perfectly aligned to fuse information from those images effectively.

In this thesis, conventional and deep learning-based optical flow methods were evaluated to assess the amount of ghosting artifacts in fused image in raw Bayer domain. Moreover, optical flow methods were compared. It was observed that the Dense Inverse Search (DIS) method outperforms other optical flow algorithms, producing a result very close to the reference frame, and almost no ghosting artifacts in fused aligned multiple frames. The CS Flow method, which is the best deep learning-based optical flow algorithm, causes slight ghosting artifacts on the moving object. On the other hand, the SIFT + FLANN method significantly solves the misalignment problem caused by global camera movement but fails for object movement. The reason for this is that the SIFT + FLANN algorithm is homography-based and homography is known as a 2D planar motion model. In other words, homography can only predict 2D motions and can be said to fail at 3D scene depth. Furthermore, since deep learning-based algorithms work on RBG images, two different methods, demosaicing and skipping mode, were used to generate RGB images from raw Bayer images. The findings showed that aligning multiple frames based on demosaicing gave better results than aligning multiple frames based on single green channel skipping mode. Because in demosaicing, missing colors are interpolated for all pixel positions in the Bayer image. In skip mode on the other hand, the image resolution is halved as one green channel drops. Since optical flow is calculated for each pixel position, it provides a more accurate alignment compared to skip mode.

In addition, the following factors were observed to affect the optical flow quality in a positive or negative way.

- Amplitude of motion.
- Discontinuity or occlusion.
- Low light conditions.

- Pre-processing steps in raw Bayer domain
- Noise

Optical flow estimation can perform reliable estimation on small motions. The main reason is that, as discussed previously, the optical flow constraint equation is valid for small motion. Optical flow algorithms fail on large motions as the optical flow constraint equation for large motion becomes invalid. Current optical flow algorithms estimate the large motion in a coarse-to-fine pyramid scheme by reducing image size at each level. Although optical flow algorithms use pyramid level, they still fail in large motions. This problem can be partially solved by increasing the pyramid level, allowing the algorithm to give better results in larger motions. But this will slow down the optical flow algorithm. Optical flow algorithms do not perform as desired when there is a discontinuity or occlusion in the scene. Therefore, optical flow algorithms have difficulty estimating where the new part of the object comes from in the frame. That results in unreliable estimates. However, Global Motion Aggregation (GMA) method aims to improve the motion estimation in the occluded region in the scene. Compared with other methods, it was observed that GMA provides better optical flow estimation in the presence of discontinuities and occlusions.In raw Bayer domain, the performance of optical flow algorithms turns out to be affected by some degradations originating from the image capture process: black level offset, lens shading/vignetting, color imbalance, and poor contrast. It is observed that the optical flow algorithms achieved remarkable overall quality improvement after applying Black Level Correction (BLC), Lens Shading Correction (LSC), White Balance (WB), and Opto-Electronic Transfer Function (OETF) to the images. Noise affects the performance of the optical flow algorithms negatively. In parallel with the increase in noise, the motion estimation is more error-prone and does not perform as desired.

For further development, the in-house dataset contains only global motion, that is, no objects that move independently of camera movement. Locally moving objects might be added to the in-house dataset. Deep-learning-based algorithms did not show the desired performance on the in-house dataset compared to their performance on the synthetically generated datasets. Furthermore, optical flow algorithms perform poorly under low light conditions. As discussed in Section 6.4, the Various Brightness Optical Flow (VBOF) dataset contains low-light noisy raw images. Optical flow algorithms may be trained on both the object added version of the in-house dataset and the VBOF dataset to improve the performance of optical flow on both real-world images and low light images. Moreover, raw Bayer images have four channels (RGGB). However, deep learning-based optical flow algorithms operate on RGB images. Therefore, deep learning-based algorithms may be modified to operate on the raw Bayer images.

REFERENCES

[1] Gibson, J. J. *The Perception Of The Visual World*. Boston: Houghton Mifflin, 1950.

[2] Horn, B. and Schunck, B. Determining Optical Flow. *Artificial Intelligence* 17 (Aug. 1981), pp. 185–203. DOI: 10.1016/0004-3702(81)90024-2.

[3] Yourgrau, W., Mandelstam, S. and Gillis, J. Variational Principles in Dynamics and Quantum Theory. 1956.

[4] Lucas, B. and Kanade, T. An Iterative Image Registration Technique with an Application to Stereo Vision (IJCAI). Vol. 81. Apr. 1981.

[5] Kroeger, T., Timofte, R., Dai, D. and Gool, L. V. Fast Optical Flow using Dense Inverse Search. *CoRR* abs/1603.03590 (2016). arXiv: 1603.03590. URL: http://arxiv.org/abs/1603.03590.

[6] Farnebäck, G. Two-Frame Motion Estimation Based on Polynomial Expansion. *SCIA*. 2003.

[7] Le Besnerais, G. and Champagnat, F. Dense Optical Flow by Iterative Local Window Registration. *IEEE International Conference on Image Processing 2005*. Vol. 1. 2005, pp. I–137. DOI: 10.1109/ICIP.2005.1529706.

[8] Sánchez, J., Meinhardt-Llopis, E. and Facciolo, G. TV-L1 Optical Flow Estimation. *Image Processing On Line* 3 (July 2013), pp. 137–150. DOI: 10.5201/ipol.2013.26.

[9] Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Smagt, P. van der, Cremers, D. and Brox, T. FlowNet: Learning Optical Flow With Convolutional Networks. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Dec. 2015.

[10] Sun, D., Yang, X., Liu, M. and Kautz, J. PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume. *CoRR* abs/1709.02371 (2017). arXiv: 1709.02371. URL: http://arxiv.org/abs/1709.02371.

[11] Bhat, G., Danelljan, M., Gool, L. V. and Timofte, R. Deep Burst Super-Resolution. *CoRR* abs/2101.10997 (2021). arXiv: 2101.10997. URL: https://arxiv.org/abs/2101.10997.

[12] Jiang, S., Campbell, D., Lu, Y., Li, H. and Hartley, R. I. Learning to Estimate Hidden Motions with Global Motion Aggregation. *CoRR* abs/2104.02409 (2021). arXiv: 2104.02409. URL: https://arxiv.org/abs/2104.02409.

[13] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. and Polosukhin, I. Attention Is All You Need. *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.

[14] Huang, Z., Shi, X., Zhang, C., Wang, Q., Cheung, K. C., Qin, H., Dai, J. and Li, H. *FlowFormer: A Transformer Architecture for Optical Flow*. 2022. DOI: 10.48550/ARXIV.2203.16194. URL: https://arxiv.org/abs/2203.16194.

[15] Butler, D. J., Wulff, J., Stanley, G. B. and Black, M. J. A Naturalistic Open Source Movie for Optical Flow Evaluation. *European Conf. on Computer Vision (ECCV)*. Ed. by A. Fitzgibbon et al. (Eds.) Part IV, LNCS 7577. Springer-Verlag, Oct. 2012, pp. 611–625.

[16] Nayar, S. K. *Edge Detection*. URL: https://cave.cs.columbia.edu/Statics/monographs/Edge%20Detection%20FPCV-2-1.pdf.

[17] Nikkanen, J. Computational Color Constancy in Mobile Imaging. Valvoja: Moncef Gabbouj. PhD thesis. Tampere: Väitöskirja : 2013.

[18] Liang, Z., Zhang, G., Huang, J. X. and Hu, Q. V. Deep Learning for Healthcare Decision Making with EMRs. *2014 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. 2014, pp. 556–559. DOI: 10.1109/BIBM.2014.6999219.

[19] Wang, G., Tan, G. W.-H., Yuan, Y., Ooi, K.-B. and Dwivedi, Y. K. Revisiting TAM2 in Behavioral Targeting Advertising: A deep Learning-Based Dual-Stage SEM-ANN analysis. *Technological Forecasting and Social Change* 175 (2022), p. 121345. ISSN: 0040-1625. DOI: https://doi.org/10.1016/j.techfore.2021.121345. URL: https://www.sciencedirect.com/science/article/pii/S0040162521007769.

[20] Lee, M.-j. and Ha, Y.-g. Autonomous Driving Control Using End-to-End Deep Learning. *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*. 2020, pp. 470–473. DOI: 10.1109/BigComp48618.2020.00-23.

[21] Singh, S. P., Kumar, A., Darbari, H., Singh, L., Rastogi, A. and Jain, S. Machine Translation Using Deep learning: An Overview. *2017 International Conference on Computer, Communications and Electronics (Comptelix)*. 2017, pp. 162–167. DOI: 10.1109/COMPTELIX.2017.8003957.

[22] Cuayáhuitl, H., Lee, D., Ryu, S., Cho, Y., Choi, S., Indurthi, S., Yu, S., Choi, H., Hwang, I. and Kim, J. Ensemble-Based Deep Reinforcement Learning for Chatbots. *Neurocomputing* 366 (2019), pp. 118–130. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2019.08.007. URL: https://www.sciencedirect.com/science/article/pii/S0925231219311269.

[23] Liao, S., Wang, J., Yu, R., Sato, K. and Cheng, Z. CNN for Situations Understanding Based on Sentiment Analysis of Twitter Data. *Procedia Computer Science* 111 (2017). The 8th International Conference on Advances in Information Technology, pp. 376–381. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2017.06.037. URL: https://www.sciencedirect.com/science/article/pii/S1877050917312103.

[24] Mustaqeem and Kwon, S. A CNN-Assisted Enhanced Audio Signal Processing for Speech Emotion Recognition. *Sensors* 20.1 (2020). ISSN: 1424-8220. DOI: 10.3390/s20010183. URL: https://www.mdpi.com/1424-8220/20/1/183.

[25] Matsumoto, T., Yokohama, T., Suzuki, H., Furukawa, R., Oshimoto, A., Shimmi, T., Matsushita, Y., Seo, T. and Chua, L. Several Image Processing Examples by CNN. *IEEE International Workshop on Cellular Neural Networks and their Applications*. 1990, pp. 100–111. DOI: 10.1109/CNNA.1990.207512.

[26] Arya, D., Maeda, H., Ghosh, S. K., Toshniwal, D., Mraz, A., Kashiyama, T. and Sekimoto, Y. Transfer Learning-based Road Damage Detection for Multiple Countries. *CoRR* abs/2008.13101 (2020). arXiv: 2008.13101. URL: https://arxiv.org/abs/2008.13101.

[27] Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. DOI: 10.1109/5.726791.

[28] Krizhevsky, A., Sutskever, I. and Hinton, G. E. ImageNet Classification with Deep Convolutional Neural Networks. *Communications of the ACM* 60 (2012), pp. 84–90.

[29] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. and Fei-Fei, L. Imagenet: A Large-Scale Hierarchical Image Database. *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[30] Simonyan, K. and Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR* abs/1409.1556 (2015).

[31] Goodfellow, I., Bengio, Y. and Courville, A. *Deep Learning*. MIT Press, 2016.

[32] Ronneberger, O., Fischer, P. and Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *CoRR* abs/1505.04597 (2015). arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597.

[33] Rumelhart, D. E., Hinton, G. E. and Williams, R. J. Learning Representations by Back-propagating Errors. *Nature* 323 (1986), pp. 533–536.

[34] Babaee, M., Li, Z. and Rigoll, G. Occlusion Handling in Tracking Multiple People Using RNN. *2018 25th IEEE International Conference on Image Processing (ICIP)*. 2018, pp. 2715–2719. DOI: 10.1109/ICIP.2018.8451140.

[35] Chien, J.-T. and Ku, Y.-C. Bayesian Recurrent Neural Network for Language Modeling. *IEEE Transactions on Neural Networks and Learning Systems* 27.2 (2016), pp. 361–374. DOI: 10.1109/TNNLS.2015.2499302.

[36] Pascanu, R., Mikolov, T. and Bengio, Y. On the Difficulty of Training Recurrent Neural Networks. *Proceedings of the 30th International Conference on Machine Learning*. Ed. by S. Dasgupta and D. McAllester. Vol. 28. Proceedings of Machine Learning Research 3. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 1310–1318. URL: https://proceedings.mlr.press/v28/pascanu13.html.

[37] Fang, X., Xu, M., Xu, S. and Zhao, P. A Deep Learning Framework for Predicting Cyber Attacks Rates. *EURASIP Journal on Information Security* 2019 (May 2019). DOI: 10.1186/s13635-019-0090-6.

[38] Liu, X., Duh, K., Liu, L. and Gao, J. *Very Deep Transformers for Neural Machine Translation*. 2020. DOI: 10.48550/ARXIV.2008.07772. URL: https://arxiv.org/abs/2008.07772.

[39] Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2018. DOI: 10.48550/ARXIV.1810.04805. URL: https://arxiv.org/abs/1810.04805.

[40] Zhai, S., Talbott, W., Srivastava, N., Huang, C., Goh, H., Zhang, R. and Susskind, J. M. An Attention Free Transformer. *CoRR* abs/2105.14103 (2021). arXiv: 2105.14103. URL: https://arxiv.org/abs/2105.14103.

[41] Harris, C. G. and Stephens, M. J. A Combined Corner and Edge Detector. *Alvey Vision Conference*. 1988.

[42] opencv. *Demosaicing*. URL: https://cave.cs.columbia.edu/Statics/monographs/Edge%20Detection%20FPCV-2-1.pdf.

[43] Lowe, D. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision* 60 (Nov. 2004), pp. 91–. DOI: 10.1023/B:VISI.0000029664.99615.94.

[44] Wang, G., Rister, B. and Cavallaro, J. Workload Analysis and Efficient OpenCL-based Implementation of SIFT Algorithm on a Smartphone. Dec. 2013. DOI: 10.1109/GlobalSIP.2013.6737002.

[45] Bay, H., Ess, A., Tuytelaars, T. and Gool, L. V. Speeded-Up Robust Features (SURF). *Comput. Vis. Image Underst.* 110 (2008), pp. 346–359.

[46] Rublee, E., Rabaud, V., Konolige, K. and Bradski, G. ORB: An Efficient Alternative to SIFT or SURF. *2011 International Conference on Computer Vision*. 2011, pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.

[47] Pham, N.-K., Morin, A. and Gros, P. Boosting of Factorial Correspondence Analysis for Image Retrieval. July 2008, pp. 224–229. ISBN: 978-1-4244-2043-8. DOI: 10.1109/CBMI.2008.4564950.

[48] Noble, F. K. Comparison of Feature of OpenCV Detectors and Feature Matchers. *2016 23rd International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*. 2016, pp. 1–6. DOI: 10.1109/M2VIP.2016.7827292.

[49] Muja, M. and Lowe, D. G. Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration. *VISAPP*. 2009.

[50] Denman, S., Fookes, C. and Sridharan, S. Improved Simultaneous Computation of Motion Detection and Optical Flow for Object Tracking. *2009 Digital Image Computing: Techniques and Applications*. 2009, pp. 175–182. DOI: 10.1109/DICTA.2009.35.

[51] Kale, K., Pawar, S. and Dhulekar, P. Moving Object Tracking Using Optical Flow and Motion Vector Estimation. *2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions)*. 2015, pp. 1–6. DOI: 10.1109/ICRITO.2015.7359323.

[52] Sevilla-Lara, L., Liao, Y., Güney, F., Jampani, V., Geiger, A. and Black, M. J. On the Integration of Optical Flow and Action Recognition. *CoRR* abs/1712.08416 (2017). arXiv: 1712.08416. URL: http://arxiv.org/abs/1712.08416.

[53] Chao, H., Gu, Y. and Napolitano, M. R. A Survey of Optical Flow Techniques for Robotics Navigation Applications. *Journal of Intelligent & Robotic Systems* 73 (2014), pp. 361–372.

[54] DeCarlo, D. and Metaxas, D. The Integration of Optical Flow and Deformable Models with Applications to Human Face Shape and Motion Estimation. *Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 1996, pp. 231–238. DOI: 10.1109/CVPR.1996.517079.

[55] Gallego, G., Rebecq, H. and Scaramuzza, D. A Unifying Contrast Maximization Framework for Event Cameras, with Applications to Motion, Depth, and Optical Flow Estimation. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3867–3876. DOI: 10.1109/CVPR.2018.00407.

[56] Luettgen, M., Clem Karl, W. and Willsky, A. Efficient Multiscale Regularization with Applications to the Computation of Optical Flow. *IEEE Transactions on Image Processing* 3.1 (1994), pp. 41–64. DOI: 10.1109/83.265979.

[57] Brox, T., Bruhn, A., Papenberg, N. and Weickert, J. High Accuracy Optical Flow Estimation Based on a Theory for Warping. Springer, 2004, pp. 25–36.

[58] Husseini, S. *A Survey of Optical Flow Techniques for Object Tracking*. Aug. 2017. URL: https://trepo.tuni.fi/handle/123456789/25188.

[59] Bergen, J. R., Anandan, P., Hanna, T. J. and Hingorani, R. Hierarchical Model-Based Motion Estimation. Springer-Verlag, 1992, pp. 237–252.

[60] Unser, M. Splines: A Perfect Fit for Signal and Image Processing. *IEEE Signal Processing Magazine* 16.6 (1999), pp. 22–38. DOI: 10.1109/79.799930.

[61] Zach, C., Pock, T. and Bischof, H. A Duality Based Approach for Realtime TV-L1 Optical Flow. *DAGM-Symposium*. 2007.

[62] Teed, Z. and Deng, J. RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. *CoRR* abs/2003.12039 (2020). arXiv: 2003.12039. URL: https://arxiv.org/abs/2003.12039.

[63] Eldesokey, A. and Felsberg, M. Normalized Convolution Upsampling for Refined Optical Flow Estimation. *CoRR* abs/2102.06979 (2021). arXiv: 2102.06979. URL: https://arxiv.org/abs/2102.06979.

[64] Shi, H., Zhou, Y., Yang, K., Yin, X. and Wang, K. CSFlow: Learning Optical Flow via Cross Strip Correlation for Autonomous Driving. *CoRR* abs/2202.00909 (2022). arXiv: 2202.00909. URL: https://arxiv.org/abs/2202.00909.

[65] Sun, D. *PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume.* 2018. URL: https://github.com/philferriere/tfoptflow.

[66] Teed, Z. and Deng, J. *RAFT: Recurrent All Pairs Field Transforms for Optical Flow.* 2020. URL: https://github.com/princeton-vl/RAFT.

[67] Eldesokey, A. and Felsberg, M. *Normalized Convolution Upsampling for Refined Optical Flow Estimation.* 2021. URL: https://github.com/abdo-eldesokey/RAFT-NCUP.

[68] Jiang, S. *Learning to Estimate Hidden Motions with Global Motion Aggregation*. 2021. URL: https://github.com/zacjiang/GMA.

[69] Shi, H. *CSFlow: Learning Optical Flow via Cross Strip Correlation for Autonomous Driving.* Feb. 2022. URL: https://github.com/MasterHow/CSFlow.

[70] Sintel. *Sintel Benchmark*. URL: http://sintel.is.tue.mpg.de/results.

[71] Ilg, E., Saikia, T., Keuper, M. and Brox, T. Occlusions, Motion and Depth Boundaries with a Generic Network for Disparity, Optical Flow or Scene Flow Estimation. *CoRR* abs/1808.01838 (2018). arXiv: 1808.01838. URL: http://arxiv.org/abs/1808.01838.

[72] Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K. and Yuille, A. L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.4 (2018), pp. 834–848. DOI: 10.1109/TPAMI.2017.2699184.

[73] Wang, X., Girshick, R. B., Gupta, A. K. and He, K. Non-local Neural Networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 7794–7803.

[74] Brown, L. G. *A Survey of Image Registration Techniques," ACM Computing Surveys (CSUR), Vol. 24, No. 4, 1992, pp. 325-376.* 1992. URL: https://www.sci.utah.edu/~gerig/CS6640-F2010/p325-brown.pdf.

[75] Du, J., Li, W., Lu, K. and Xiao, B. An overview of Multi-modal Medical Image Fusion. *Neurocomputing* 215 (2016). SI: Stereo Data, pp. 3–20. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2015.07.160. URL: https://www.sciencedirect.com/science/article/pii/S092523121630649X.

[76] Zhang, Y., Bai, X. and Wang, T. Boundary Finding Based Multi-Focus Image Fusion through Multi-Scale Morphological Focus-Measure. *Information Fusion* 35 (2017), pp. 81–101. ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2016.09.006. URL: https://www.sciencedirect.com/science/article/pii/S1566253516300793.

[77] Ma, K., Zeng, K. and Wang, Z. Perceptual Quality Assessment for Multi-Exposure Image Fusion. *IEEE Transactions on Image Processing* 24.11 (2015), pp. 3345–3356. DOI: 10.1109/TIP.2015.2442920.

[78] Geiger, A., Lenz, P. and Urtasun, R. Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite. *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[79] Menze, M. and Geiger, A. Object Scene Flow for Autonomous Vehicles. *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[80] Zhang, M., Zheng, Y. and Lu, F. Optical Flow in the Dark. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021), pp. 1–1. DOI: 10.1109/TPAMI.2021.3130302.

[81] Mayer, N., Ilg, E., Häusser, P., Fischer, P., Cremers, D., Dosovitskiy, A. and Brox, T. A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:1512.02134. 2016. URL: http://lmb.informatik.uni-freiburg.de/Publications/2016/MIFDB16.

[82] Barron, J. L., Fleet, D. J. and Beauchemin, S. S. Performance of Optical Flow Techniques. *International Journal of Computer Vision* 12 (1992), pp. 43–77.

[83] Horé, A. and Ziou, D. Image Quality Metrics: PSNR vs. SSIM. *2010 20th International Conference on Pattern Recognition* (2010), pp. 2366–2369.

[84] Farias, M. Video Quality Metrics. Feb. 2010. ISBN: 978-953-7619-70-1. DOI: 10.5772/8038.

[85] Yuhas, R. H., Goetz, A. F. H. and Boardman, J. W. Discrimination Mmong Semi-Arid Landscape Endmembers Using the Spectral Angle Mapper (SAM) algorithm. 1992.

[86] Wang, Z. and Bovik, A. A Universal Image Quality Index. *IEEE Signal Processing Letters* 9.3 (2002), pp. 81–84. DOI: 10.1109/97.995823.

[87] Rodgers, J. L. and Nicewander, W. A. Thirteen Ways to Look at the Correlation Coefficient. *The American Statistician* 42.1 (1988), pp. 59–66. ISSN: 00031305. URL: http://www.jstor.org/stable/2685263 (visited on 08/06/2022).

[88] URL: https://www.gsmarena.com/xiaomi_mi_11_pro-10816.php.

APPENDIX A: CLEAN PASS RESULTS OF THE SINTEL DATASET

Table A.1. *Results of clean pass using deep learning-based optical flow estimation methods. Where red highlighted values indicate the highest error, while green highlighted values indicate the lowest error in each column*

scene name	GMA							RAFT							RAFT-NCUP							CSFLOW							PWC-NET						
	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+
alley_1	0.18	0.14	0.92	0.16	96.63	3.21	0.16	0.18	0.15	0.92	0.15	96.60	3.23	0.18	0.21	0.18	1.01	0.20	95.93	3.83	0.24	0.18	0.15	0.91	0.12	96.59	3.23	0.18	0.25	0.20	1.41	0.45	95.36	4.27	0.37
alley_2	0.17	0.17	1.05	1.14	98.51	1.22	0.27	0.17	0.18	1.03	0.64	98.47	1.23	0.29	0.20	0.20	1.14	0.44	98.03	1.59	0.38	0.17	0.17	1.04	0.43	98.50	1.23	0.26	0.28	0.25	1.75	1.53	97.20	1.94	0.86
ambush_2	4.85	3.57	1.54	4.92	77.58	15.85	6.57	4.53	2.20	1.46	4.59	76.60	16.52	6.88	6.12	2.19	2.04	6.32	70.75	19.37	9.88	5.52	3.59	1.97	5.43	77.65	15.76	6.59	7.89	7.15	3.61	8.70	61.14	20.42	18.43
ambush_4	5.38	3.64	5.66	13.94	66.42	16.71	16.87	9.56	7.96	12.51	17.79	65.41	16.02	18.57	10.01	5.93	12.19	22.63	62.10	16.85	21.04	8.20	8.32	12.31	13.58	66.17	17.24	16.60	11.30	6.56	7.61	31.44	59.02	16.17	24.81
ambush_5	1.97	1.66	2.64	6.34	82.94	11.62	5.44	2.32	1.79	2.76	6.72	81.94	11.99	6.07	3.06	2.32	4.50	6.92	74.85	16.46	8.68	2.23	1.63	2.77	6.78	81.88	12.37	5.75	5.89	3.71	8.38	9.35	68.44	15.90	15.66
ambush_6	3.44	6.72	1.43	5.76	77.2	15.3	7.50	3.49	6.85	1.47	5.96	77.10	15.45	7.45	3.85	7.18	1.74	6.59	72.42	18.6	8.97	3.40	6.51	1.44	5.65	77.54	15.33	7.13	6.15	9.04	2.10	9.99	69.03	18.93	12.04
ambush_7	0.30	0.20	1.89	2.39	97.58	1.89	0.53	0.27	0.19	2.11	1.60	97.62	1.92	0.46	0.33	0.23	2.23	1.75	96.98	2.42	0.60	0.26	0.19	2.10	1.69	97.68	1.89	0.43	0.57	0.50	2.99	3.38	95.50	3.17	1.33
bamboo_1	0.40	0.33	7.93	4.97	93.25	5.83	0.92	0.42	0.34	8.28	5.19	92.89	6.18	0.92	0.45	0.37	8.03	5.31	91.58	7.50	0.92	0.40	0.33	7.38	4.66	92.78	6.43	0.80	0.45	0.39	6.83	4.01	91.11	8.01	0.89
bamboo_2	0.60	0.30	7.86	14.45	93.50	4.94	1.56	0.60	0.31	7.51	13.11	93.55	4.86	1.59	0.64	0.35	7.42	14.09	92.57	5.64	1.79	0.59	0.32	6.86	12.19	93.56	4.87	1.57	0.81	0.41	7.65	14.91	91.67	6.07	2.27
bandage_1	0.34	0.28	0.98	7.87	95.12	4.02	0.86	0.35	0.30	1.04	7.86	94.62	4.37	1.00	0.42	0.35	1.13	7.70	93.36	5.43	1.20	0.36	0.30	0.95	8.16	95.04	3.96	1.00	0.51	0.44	1.35	8.09	92.28	6.00	1.72
bandage_2	0.24	0.21	0.98	0.02	96.10	3.52	0.38	0.24	0.21	1.05	0.04	96.05	3.60	0.35	0.27	0.24	1.12	0.04	95.22	4.38	0.40	0.23	0.21	1.09	0.04	96.07	3.60	0.33	0.32	0.28	1.32	0.56	93.88	5.47	0.65
cave_2	3.75	1.60	1.88	5.60	83.08	11.66	5.27	3.55	1.64	1.86	5.44	83.14	11.64	5.22	3.82	1.71	2.11	5.75	79.48	13.78	6.74	3.61	1.59	1.92	5.46	82.97	11.98	5.05	5.92	3.29	4.37	8.47	71.52	15.91	12.57
cave_4	2.19	1.63	1.43	26.51	80.16	13.87	5.97	2.13	1.60	1.45	24.03	79.83	14.20	5.98	2.17	1.93	1.55	25.9	77.77	15.60	6.63	2.02	1.59	1.41	23.44	80.11	14.18	5.71	3.19	2.29	2.04	30.77	73.99	16.76	9.25
market_2	0.39	0.23	3.22	6.89	94.27	4.58	1.15	0.41	0.26	3.32	5.97	93.74	5.06	1.20	0.46	0.28	3.65	6.48	92.83	5.69	1.48	0.43	0.26	3.08	6.36	93.93	4.81	1.26	0.54	0.34	4.39	7.77	92.50	5.58	1.92
market_5	4.73	2.56	2.16	10.68	74.77	14.68	10.54	4.77	2.58	2.15	10.76	75.29	14.64	10.06	5.70	2.74	3.33	11.82	70.70	16.47	12.83	4.78	2.28	2.48	10.25	75.23	14.52	10.25	10.73	4.25	6.91	21.45	59.69	19.17	21.15
market_6	1.78	0.95	1.30	16.65	77.51	15.80	6.69	1.96	1.00	1.48	16.79	76.33	16.37	7.30	2.20	1.15	1.84	16.94	71.14	20.38	8.48	1.67	0.99	1.43	12.09	77.05	16.59	6.36	4.03	2.40	2.11	26.19	62.40	23.48	14.13
mountain_1	0.20	0.13	0.60	3.97	97.84	1.84	0.32	0.20	0.12	0.63	4.26	97.99	1.72	0.29	0.23	0.14	0.64	4.60	97.51	2.16	0.33	0.19	0.12	0.53	4.33	98.20	1.56	0.24	0.33	0.17	0.96	7.23	96.73	2.58	0.69
shaman_2	0.21	0.21	1.82	0.00	97.49	2.42	0.09	0.21	0.21	1.85	0.00	97.54	2.38	0.09	0.23	0.23	1.81	0.00	97.30	2.63	0.07	0.21	0.21	1.82	0.00	97.53	2.39	0.09	0.28	0.27	1.82	0.00	96.55	3.37	0.07
shaman_3	0.16	0.15	0.41	0.00	99.05	0.88	0.08	0.16	0.16	0.39	0.00	98.86	1.05	0.09	0.19	0.19	0.54	0.00	98.49	1.39	0.12	0.17	0.16	0.44	0.00	98.82	1.07	0.11	0.27	0.26	0.85	0	96.26	3.4	0.34
sleeping_1	0.12	0.12	0.00	0.00	99.94	0.06	0.00	0.11	0.11	0.00	0.00	99.91	0.09	0.00	0.14	0.14	0.00	0.00	99.88	0.12	0.00	0.11	0.11	0.00	0.00	99.94	0.06	0.00	0.16	0.16	0.00	0.00	99.58	0.41	0.01
sleeping_2	0.11	0.11	0.00	0.00	99.68	0.32	0.00	0.11	0.11	0.00	0.00	99.7	0.30	0.00	0.13	0.13	0.00	0.00	99.68	0.32	0.00	0.11	0.11	0.00	0.00	99.74	0.26	0.00	0.15	0.15	0.00	0.00	99.15	0.85	0.00
temple_2	1.29	0.45	1.00	18.00	91.76	5.44	2.80	1.63	0.48	1.14	19.79	91.45	5.75	2.8	1.69	0.54	1.09	21.45	89.66	7.05	3.29	1.14	0.46	0.74	15.77	92.08	5.22	2.70	1.79	0.67	0.96	18.33	87.55	7.82	4.63
temple_3	2.21	3.97	2.83	12.94	82.80	10.81	6.39	2.52	5.19	3.06	12.80	81.64	11.26	7.11	3.26	3.34	3.74	16.27	77.05	13.57	9.38	2.60	5.43	3.60	16.19	82.04	11.42	6.54	6.69	3.58	4.87	18.94	65.89	15.74	18.37
Averaged Error	1.31	1.02	2.14	6.99	90.43	6.53	3.04	1.48	1.20	2.40	6.94	90.16	6.67	3.17	1.69	1.13	2.63	7.63	88.16	7.93	3.91	1.41	1.22	2.34	6.53	90.33	6.68	2.99	2.60	1.61	3.19	9.60	84.95	8.82	6.23

Table A.2. *Results of clean pass using conventional optical flow estimation methods and global motion estimation-based method. Where red highlighted values indicate the highest error, while green highlighted values indicate the lowest error in each column*

scene name	Farneback							TV-L1							Iterative Lucas-Kanade (ILK)							Dense Inverse Search (DIS)							MiHAI (SIFT + FLANN)						
	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+	AEPE	s0-10	s10-40	s40+	e1	e1-5	e5+
alley_1	0.97	0.79	4.65	3.41	82.92	11.35	5.73	0.50	0.38	2.64	1.02	90.92	7.06	2.01	0.76	0.63	3.89	2.68	87.6	8.75	3.66	0.44	0.39	1.95	0.85	91.94	6.52	1.54	1.63	0.99	10.94	4.36	75.53	14.14	10.33
alley_2	1.54	1.18	5.30	1.75	74.44	14.17	11.39	0.37	0.30	5.08	1.74	96.39	2.05	1.56	0.88	0.60	4.98	1.85	91.32	3.97	4.71	0.48	0.40	3.56	1.76	94.94	3.05	2.02	0.88	0.82	6.83	1.76	84.04	13.71	2.25
ambush_2	60.62	8.55	23.04	73.69	1.98	2.89	95.13	37.96	12.09	10.29	46.57	16.00	11.38	72.62	37.48	21.28	19.32	41.64	29.36	11.37	59.28	15.52	12.81	7.71	17.35	55.35	15.41	29.24	30.29	25.79	30.38	33.22	13.39	23.06	63.56
ambush_4	28.09	8.64	16.86	76.96	21.21	9.20	69.59	21.42	9.48	13.23	64.12	40.49	11.96	47.55	32.61	22.34	32.01	62.57	35.52	10.01	54.48	29.55	17.77	36.27	59.62	50.32	10.62	39.06	22.47	11.50	13.45	75.07	54.54	6.85	38.61
ambush_5	18.51	3.30	12.03	43.09	31.00	22.12	46.88	15.22	4.69	7.92	34.07	42.71	14.51	42.78	16.90	7.88	14.34	31.82	48.55	15.83	35.62	9.05	4.18	7.38	17.60	63.23	16.08	20.69	18.36	5.61	17.28	44.13	33.03	21.93	45.04
ambush_6	28.84	8.62	15.91	48.86	17.32	9.32	73.36	17.54	12.54	7.24	32.80	27.45	20.75	51.79	23.10	19.56	15.18	34.59	33.11	15.24	51.64	14.10	12.23	4.86	25.64	59.42	14.02	26.57	22.37	23.24	12.09	39.89	26.24	22.74	51.02
ambush_7	3.03	0.98	7.55	16.37	86.98	4.18	8.84	3.51	2.40	5.38	14.28	70.56	13.53	15.91	4.16	3.95	5.66	9.00	80.68	5.10	14.22	1.11	0.99	5.59	6.13	93.12	3.45	3.44	5.21	5.65	7.90	13.70	64.11	13.09	22.80
bamboo_1	0.69	0.60	9.24	5.02	83.84	13.43	2.73	0.50	0.41	10.75	5.11	89.69	8.76	1.55	0.72	0.63	9.33	5.03	83.87	13.13	2.99	0.54	0.46	9.05	5.02	88.08	10.33	1.59	0.89	0.77	14.21	5.15	81.30	14.42	4.27
bamboo_2	1.72	0.70	17.11	30.55	84.22	10.29	5.49	1.35	0.38	17.24	29.25	91.3	4.76	3.94	1.80	1.02	14.25	26.57	83.89	9.39	6.72	1.27	0.56	13.25	26.83	88.91	7.16	3.92	1.75	0.59	21.22	30.52	86.16	9.62	4.22
bandage_1	1.73	1.14	4.96	8.36	74.69	13.78	11.53	0.97	0.75	3.05	8.28	81.72	12.36	5.92	1.29	1.03	3.25	8.74	79.49	12.77	7.74	0.73	0.63	1.86	8.14	88.21	7.95	3.84	3.52	2.16	13.57	7.83	50.7	23.59	25.71
bandage_2	0.83	0.67	3.81	0.76	83.38	11.89	4.73	0.54	0.44	3.23	0.61	88.46	9.34	2.19	0.72	0.65	2.64	0.63	86.23	9.82	3.95	0.46	0.42	1.74	0.08	90.86	7.26	1.88	2.40	1.69	11.22	0.89	56.31	28.91	14.78
cave_2	34.24	3.54	12.43	44.97	23.08	9.69	67.23	29.94	4.74	10.28	39.37	26.9	10.23	62.87	18.93	6.80	11.07	24.42	49.06	10.35	40.59	11.58	4.23	6.34	16.11	66.16	11.28	22.56	20.63	25.61	25.23	23.88	28.96	24.22	46.82
cave_4	8.20	4.18	7.13	53.69	46.80	17.91	35.29	6.69	4.23	5.60	56.45	53.88	20.98	25.14	7.29	5.54	5.48	49.92	58.03	15.51	26.46	5.53	3.39	3.47	49.06	69.45	14.67	15.89	10.86	7.77	10.44	58.50	31.45	25.94	42.62
market_2	2.25	0.91	12.22	18.67	80.95	7.17	11.88	1.16	0.62	7.01	14.63	86.44	7.37	6.19	2.06	1.16	9.89	16.75	81.07	9.05	9.87	1.13	0.71	6.27	12.89	87.48	7.35	5.17	2.46	0.48	16.51	20.33	84.11	3.63	12.25
market_5	33.79	7.64	16.3	72.67	13.81	9.82	76.36	23.8	9.72	7.30	57.94	38.65	17.12	44.22	25.99	13.61	15.29	52.63	37.99	14.22	47.79	17.85	7.52	11.09	39.96	53.50	15.53	30.98	30.83	19.76	15.56	62.64	21.68	29.27	49.05
market_6	11.92	4.78	10.03	55.29	32.91	23.24	43.84	8.91	2.28	6.32	49.88	46.22	24.54	29.24	11.60	3.96	9.79	47.49	45.81	20.79	33.41	5.91	2.61	3.84	37.37	58.91	21.84	19.26	12.83	6.55	9.14	54.96	16.59	31.92	51.49
mountain_1	2.51	1.23	9.94	18.71	64.03	21.72	14.24	0.84	0.25	3.96	12.67	89.35	6.55	4.10	1.30	0.39	6.37	22.59	90.45	5.07	4.48	0.43	0.20	1.48	7.74	94.50	4.12	1.38	2.40	1.15	9.51	17.91	62.41	27.85	9.75
shaman_2	0.47	0.47	1.96	0.00	86.36	13.11	0.52	0.28	0.28	2.23	0	94.22	5.60	0.18	0.32	0.31	1.82	0.00	93.94	5.61	0.45	0.24	0.24	1.68	0.00	96.16	3.66	0.17	1.91	1.90	3.20	0.00	40.29	52.88	6.83
shaman_3	0.79	0.74	2.95	0.00	82.61	13.76	3.64	0.36	0.35	1.17	0.00	92.44	6.57	0.98	0.59	0.58	1.73	0.00	89.16	8.04	2.80	0.36	0.35	0.88	0.00	93.40	5.55	1.05	1.43	1.32	5.98	0.00	60.47	34.73	4.80
sleeping_1	1.38	1.38	0.00	0.00	60.82	34.34	4.84	0.16	0.16	0.00	0.00	99.64	0.36	0.00	0.23	0.23	0.00	0.00	97.30	2.50	0.21	0.17	0.17	0.00	0.00	99	0.97	0.03	0.33	0.33	0.00	0.00	94.27	5.73	0.00
sleeping_2	0.28	0.28	0.00	0.00	92.77	6.87	0.36	0.11	0.11	0.00	0.00	99.44	0.56	0.00	0.16	0.16	0.00	0.00	97.5	2.44	0.06	0.12	0.12	0.00	0.00	99.05	0.94	0.01	0.46	0.46	0.00	0.00	91.57	8.43	0.00
temple_2	6.81	2.36	4.87	42.53	56.77	14.38	28.86	4.39	1.19	2.16	38.66	75.11	11.96	12.92	6.07	4.04	4.8	32.78	72.68	9.88	17.44	3.28	1.94	2.56	26.56	81.28	9.43	9.29	7.50	3.76	4.57	43.72	26.16	42.46	31.38
temple_3	35.91	5.88	17.64	63.56	4.72	6.74	88.54	23.67	4.22	9.34	49.22	31.80	14.10	54.10	29.87	11.76	20.54	53.80	39.39	9.24	51.37	12.60	7.88	14.41	31.11	62.87	11.00	26.13	46.44	22.37	38.46	61.89	2.98	11.82	85.20
Averaged Error	10.33	2.55	8.69	26.71	59.39	13.46	27.14	7.43	2.52	5.88	22.43	71.55	10.06	18.39	8.23	4.47	8.37	21.10	72.14	9.62	18.24	4.86	2.75	5.85	15.79	81.31	8.58	10.10	9.69	6.38	12.51	24.43	53.66	21.39	24.95