Contribution à la modélisation d'objets extrudés : modèles théoriques et application à la C.F.A.O.

Christian Minich

Laboratoire de Recherche en Informatique de Metz

THESE

présentée à

L'UNIVERSITE DE METZ

en vue de l'obtention du grade de

DOCTEUR DE L'UNIVERSITE DE METZ
(mention Sciences)

SPECIALITE INFORMATIQUE

par

Christian MINICH

"CONTRIBUTION A LA MODELISATION D'OBJETS EXTRUDES: MODELES THEORIQUES ET APPLICATION A LA C.F.A.O."

Soutenue le 5 février 1991 devant la commission
d'examen composée de Messieurs

P. COIFFET (Rapporteur)	Directeur de recherche au CNRS
Y. GARDAN (Directeur de thèse)	Professeur à l'université de Metz
J.P. GILLET (Examinateur)	Responsable systèmes d'information, informatique technique, méthodes centrales (Renault)
J.P. HATON (Rapporteur)	Professeur à l'université de Nancy
M. POTIER-FERRY (Examinateur)	Professeur à l'université de Metz

A mes parents

REMERCIEMENTS

Maintenant que j'ai écrit ce mémoire, je comprends beaucoup mieux la valeur des sempiternels remerciements que l'on trouve toujours dans les premières pages.

C'est donc le plus chaleureusement et le plus sincèrement du monde que je tiens à exprimer ma gratitude aux personnes suivantes:

A Yvon Gardan (ou Papa ou l'affreux rouquin suivant la manière dont on le connaît) dont j'aimerais que tout le monde sache combien il est généreux.

Aux autres membres du jury qui ont pris le temps d'une lecture attentive et critique que l'on apprécie d'autant plus connaissant leur emploi du temps.

Aux personnes que j'ai interrogées, qui m'ont répondu avec patience et dont j'ai beaucoup apprécié qu'elles se mettent à ma portée. Je pense tout particulièrement à Mr Roux, Professeur de mathématiques et à Mr Schweitzer qui enseigne les techniques de fabrication.

A tous les membres du laboratoire, qui ont lu mon manuscrit, qui ont éclairé ma lanterne, qui ont subi les répétitions ou qui se sont simplement montrés sympatiques.

A Myriam, qui m'a déjà beaucoup aidé et qui continuera.

Merci à tous.

TABLE DES MATIERES

AVERTISSEMENT

Nous adoptons dans ce qui suit un certain nombre de conventions résumées ici:

- le produit vectoriel est représenté par le signe ^.

- le produit scalaire est représenté par le signe x.

- les vecteurs ne sont pas surmontés d'une flèche, le contexte suffisant à les distinguer des scalaires.

- la norme d'un vecteur est représentée par un parenthésage avec des doubles barres verticales: on notera donc || V ||.

- pour soulager l'écriture, et à moins que la clarté n'en souffre, on omettra souvent d'indiquer les paramètres d'une fonction.

- dans le texte, le signe d'union apparaît le plus souvent sous la forme d'un plus ("+"). Dans les illustrations, par contre, le signe conventionnel est utilisé.
  De même, intersection, différence et l'ensemble vide sont respectivement matérialisés par &, - et {}.

INTRODUCTION

INTRODUCTION

Il n'y a pas si longtemps, les concepteurs des avions Boeing dessinaient à l'échelle 1, dans des salles immenses et à même le sol, les plans des ailes de leurs avions.

Aujourd'hui, grâce aux systèmes de C.A.O., ils décrivent ces ailes d'une manière plus fine, testent leur résistance et leur déformabilité sans quitter leur bureau.

Dans le même temps, on conçoit des moteurs, des raffineries, des bâtiments, on simule le déplacement de robots, on vérifie le bon fonctionnement de mécanismes, on calcule des coefficients de pénétration dans l'air... la liste peut être allongée à volonté.

Cette variété des applications C.A.O. a conduit à une multitude d'outils de description: description de formes, d'assemblages, de mécanismes, de dialogue... à nouveau, il paraît illusoire de vouloir en établir une liste complète.

Le propos de ce mémoire est d'étudier un outil de description de formes, l'opérateur d'extrusion, au travers de quelques étapes de la production: ainsi, nous serons amenés à en définir la portée et le mode d'emploi, afin que l'utilisateur du système de C.A.O. en fasse le meilleur usage.

Nous nous placerons ensuite dans un contexte plus technique pour étudier dans quelles limites l'opérateur théorique d'extrusion constitue effectivement un outil de modélisation.

A titre d'illustration et afin de fouiller plus précisément encore certains problèmes, nous citerons quelques exemples de réalisation.

Comme on le verra, l'extrusion peut également être rapprochée des techniques de fabrication. La dernière partie fera donc état de ce lien et l'on profitera de cette incursion dans le domaine de la F.A.O. pour se livrer à quelques réflexions, notamment sur les possibilités d'intégrer conception et fabrication.

CHAPITRE I

1. DEFINITION ET PORTEE DE L'OPERATEUR D'EXTRUSION

1.1. Généralités

Le petit Larousse donne la définition suivante de l'extrusion: "filage à chaud de différents métaux poussés par une presse à travers une filière présentant un profil donné".

Cette technique a inspiré le dialogue d'une opération de modélisation consistant à déplacer un contour le long d'une trajectoire rectiligne pour définir un volume constitué de l'ensemble des points balayés. On crée ainsi un objet de type prisme (une surface prismatique est la surface engendrée par une droite de direction constante qui se déplace en s'appuyant constamment sur le pourtour d'un polygone plan).

Cette méthode, aisée à réaliser d'un point de vue informatique, présente en outre l'avantage d'un dialogue très simple, essentiellement bi-dimensionnel.

La technique du balayage de l'espace (sweeping) peut toutefois être considérablement étendue et être formulée ainsi: " un objet construit par extrusion (au sens informatique cette fois, non plus au sens sidérurgique) est constitué de l'ensemble des points touchés par un ***générateur*** pendant son ***déplacement*** (ou ***mouvement***) ".

Le générateur peut accessoirement être un point, mais consiste le plus souvent en une ligne (segment ou courbe), en un objet à deux dimensions (polygone, contour plan, surface), voire quelquefois en un volume.

Le mouvement regroupe les notions de *positionnement*, *d'orientation* et de *déformation* du générateur: il peut être exprimé de nombreuses manières, dépendant de l'application, de la complexité du dialogue à mettre en oeuvre et du nombre de contraintes imposées.

Dans cette approche, en parlant d'*extrusion*, on commet un abus de langage puisque le terme s'applique a priori à l'objet engendré par une section plane subissant une trajectoire rectiligne, perpendiculaire au plan de la section. Il faudrait, pour être rigoureux, employer le terme d'extrusion *généralisée* ou l'anglais *sweeping* (*balayage*). Pourtant, afin de soulager l'écriture et d'éviter les anglicismes, nous avons préféré parler d'extrusion.

Il est important de remarquer combien il est facile de construire par extrusion des objets non homogènes, en ce sens qu'ils n'ont pas partout *la même dimension* (fig 1.1.a). Suivant le contexte, ce type de construction peut ou ne pas être admis: dans un logiciel artistique,

où l'on privilégie l'aspect visuel sans soucis de la réalité physique des choses, rien ne s'y oppose. Dans une approche plus technique comme, dans le cas qui nous intéresse, un système de C.F.A.O. appliqué à la mécanique, ce n'est pas acceptable. Il apparaît donc que, quel que soit le type de structure de données utilisé pour stocker l'objet, il faudra se préoccuper d'assurer sa validité, autrement dit assurer que l'objet soit tel qu'il vérifie:

$$\overline{\dot{A}} = A$$

où ° désigne l'intérieur
et - désigne l'adhérence.

Cette propriété caractérise un ensemble de points dit *régulier*.

Lorsque cet ensemble est censé constituer un volume, nous imposons en plus que la frontière de l'objet soit une *variété de dimension 2* [MAN 88].

Une surface possède cette propriété si chacun de ses points a un voisinage topologiquement équivalent à un disque ouvert de E^2 (l'espace euclidien plan).

Un voisinage est topologiquement équivalent à un disque ouvert de E^2 s'il est homéomorphe à $\mathbf{R}^2$ [AGO 76].

Lorsque cet ensemble de points est censé constituer une surface, il doit être une variété *bornée* de dimension 2, c'est-à-dire que chaque point a un voisinage topologiquement équivalent à un disque ou à un demi-disque de E^2 (c'est-à-dire la partie du disque se trouvant d'un coté d'une droite passant par l'origine).

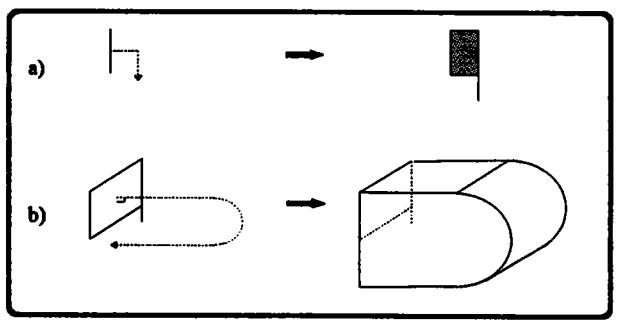


figure 1.1: a) un générateur homogène engendre un objet non homogène
b) un générateur non homogène engendre un volume homogène

La validité apparaît d'autant plus difficile à vérifier que l'homogéneité du résultat ne dépend que dans une faible mesure de celle du générateur (fig 1.1.b): tout au plus peut-on dire qu'*en général*:

- si le générateur est de dimension 1 alors l'objet construit est de dimension 2 (fig 1.2.a)
- si le générateur est de dimension 2 ou 3 alors l'objet construit est de dimension 3 (fig 1.2.b et 1.2.c).

c'est-à-dire que l'extrusion engendre dans la plupart des cas un objet d'une dimension supérieure au générateur.

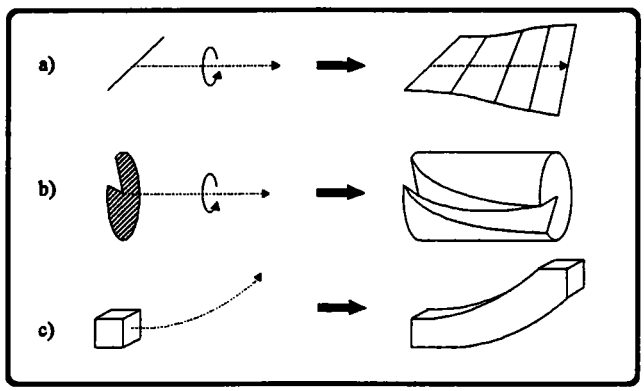


figure 1.2

La fonction d'extrusion constitue donc un puissant outil de conception: en théorie, tout objet peut être représenté par une extrusion; en pratique, il est clair cependant, ne serait-ce qu'à cause des difficultés de dialogue dans les cas extrêmes, qu'un système de C.A.O. ne peut s'en contenter.

Cette variété des possibilités de modélisation mène également à la remarque suivante: il serait sans doute irréaliste d'essayer de produire un unique dialogue et un unique algorithme d'extrusion: c'est irréalisable et quand même ça le serait, il y a tout à gagner en tirant avantage des spécificités de chaque cas, que ce soit au bénéfice de l'interactivité ou des calculs: la vocation d'un système, en particulier d'un système industriel, doit être aussi de rester pragmatique, c'est-à-dire, entre autres, de ménager l'opérateur tant au niveau du nombre de ses interventions qu'à celui des temps de réponse.

Nous terminons avec les domaines d'application des techniques d'extrusion: nous avons déjà évoqué celui de la modélisation géométrique (bâtiment, tuyauterie, pièces mécaniques) dont l'objectif est d'aboutir à une maquette informatique d'un objet. Celle-ci n'est pourtant pas toujours une fin en soi: il arrive qu'elle ne soit que le support d'autres applications, dans lesquelles l'extrusion a encore son importance:

- en robotique, elle sert à évaluer le domaine atteignable par l'outil situé à l'extrémité d'un bras articulé. On procède ainsi: le volume V_0 atteignable par l'outil seul (une

pince par exemple) n'est fonction que de ce dernier. Le volume V_n atteignable par le dernier segment du bras (celui auquel est attaché l'outil) est obtenu par déplacement de V_0 selon une trajectoire dépendant des degrés de liberté du segment par rapport à son précédent; de proche en proche, l'on détermine ainsi le volume accessible au bras complet.

- elle a également sa place en simulation d'usinage: en soustrayant à la maquette du brut le volume balayé par l'outil et en comparant le résultat au modèle de la pièce souhaitée, on peut valider (ou non) la trajectoire d'outil calculée par l'étape de préparation à la fabrication (process planning). Un exemple en est donné en 2.4.4.4.
- la bonne marche d'un mécanisme peut également être vérifiée, en étudiant les interférences de la zone balayée par la pièce en mouvement avec le reste du mécanisme. L'on évite ainsi de calculer les intersections du support avec un échantillonage des positions de la pièce en mouvement.

  La prévision des collisions entre plusieurs pièces en mouvement demande une analyse plus subtile car on ne peut affirmer qu'il y aura contact simplement parce que les volumes balayés se touchent: pour cela, il faut en plus que les pièces mobiles soient au même endroit au même instant.

1.2. Définition du déplacement

Jusqu'à présent, nous sommes restés très vague dans la définition du déplacement. Dans ce paragraphe, nous revenons donc d'abord sur quelques uns des moyens traditionnels d'exprimer la position, l'orientation et la déformation du générateur au fil de son mouvement. Ceux-ci seront ensuite illustrés par des exemples (1.2.2.)

1.2.1. Les déplacements associés aux constructions classiques

La majorité des opérations d'extrusion se pratique avec un *générateur* plan, de type contour, se déplaçant sans *déformation*. La *position* du générateur au fil du mouvement est définie par une trajectoire vis-à-vis de laquelle il conserve une *orientation* constante dans un sens que nous définissons ci-dessous (fig 1.3)

Le contour engendre alors une surface tubulaire mais on peut également convenir que l'on définit ainsi un volume bordé par la surface tubulaire et, si cette dernière n'est pas fermée, par le contour à ses positions initiale et finale. Dans ce qui suit, nous nous plaçons dans ce deuxième cas.

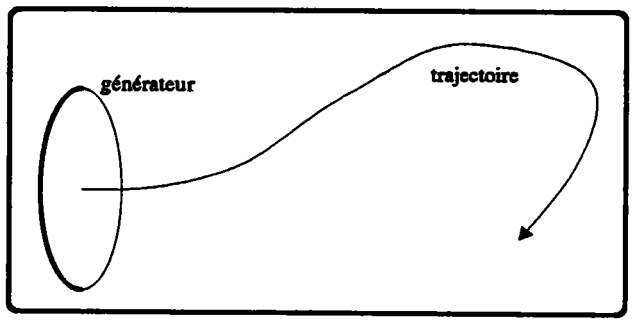


figure 1.3

La trajectoire, contrairement à ce que suggèrent les lignes qui précèdent, n'est pas rattachée au générateur lui-même mais à l'un de ses points caractéristiques: en effet, on n'engendre pas le même objet suivant que l'on affecte une même trajectoire à un point ou à un autre (fig 1.4), sauf cas particuliers (par exemple une translation).

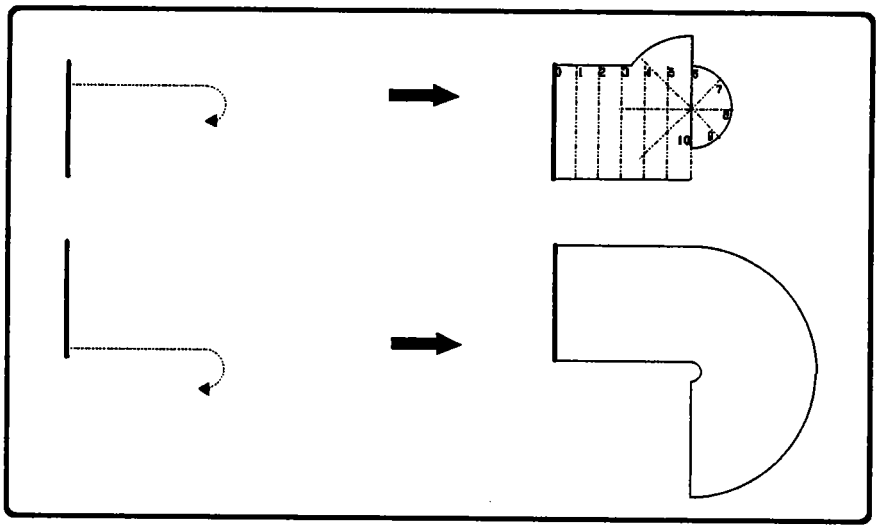


figure 1.4: une même trajectoire appliquée à des points différents du générateur

La vue est prise de coté pour simplifier le dessin.

Le point choisi est en général l'origine d'un repère local dans lequel est exprimé le générateur.

On peut alors clarifier l'expression "orientation constante": elle signifie que la normale au plan du contour forme un angle constant, strictement compris entre -90 et +90 degrés, avec la tangente à la trajectoire. Ceci n'exclut donc pas un mouvement rotatif instantané du contour (vrillage); à l'instant t, correspondant à la position O_t du point caractéristique, cette rotation doit s'effectuer autour d'un axe passant par O_t et perpendiculaire au plan du contour, de façon à ce que le contour ne quitte pas son plan (fig 1.5). Nous verrons en 1.2.2.1. un exemple regroupant ces notions et montrant entre autres comment faire un lien entre le mouvement rotatif et la torsion de la courbe.

Nous soulignons ce point afin qu'il soit clair que *l'orientation constante* n'est pas relative au repère de référence mais bien à la trajectoire.

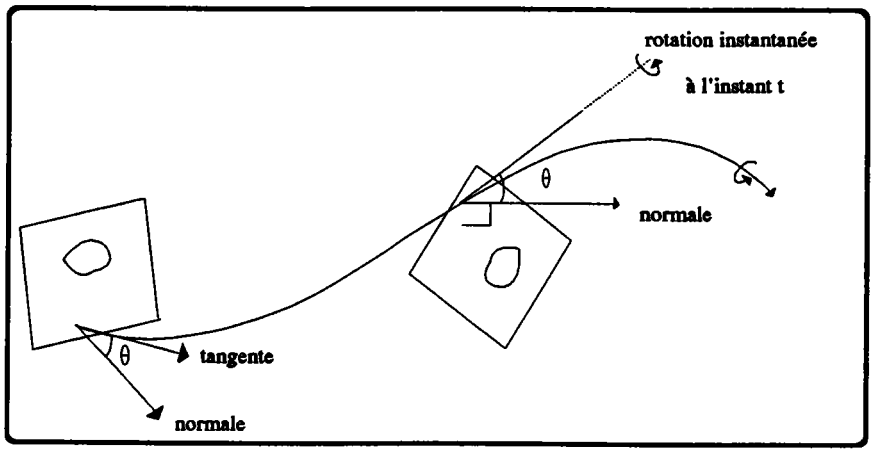


figure 1.5

Un cas particulier de ce type de construction est constitué de la famille des objets dont le générateur reste perpendiculaire à la trajectoire et que l'on regroupe sous la dénomination de **cylindres généralisés**. Parmi ceux là, on distingue les prismes dont l'arête est perpendiculaire à la base, c'est-à-dire les prismes droits. On y inclut aussi les objets de révolution dont l'axe est dans un même plan que le générateur (ou premier méridien).

Les cylindres généralisés ont la propriété que toute coupe par un plan perpendiculaire à la trajectoire est une courbe égale au générateur lui-même, dans la limite où l'objet ne se recoupe pas.

Les autres objets construits avec un générateur plan et une orientation constante mais oblique peuvent, sous certaines conditions, être rapprochés des cylindres généralisés. Nous voyons sur l'illustration ci-dessous (fig 1.6.a) un cas simple pour lequel on peut, aux extrémités près, engendrer un même objet avec un générateur oblique et un autre, perpendiculaire. Ce dernier est obtenu par projection orthogonale du générateur sur un plan perpendiculaire à la trajectoire.

Une situation du même ordre survient lorsque l'on construit une surface de révolution avec un axe strictement parallèle au plan du contour: bien que ce dernier se déplace alors sans être perpendiculaire à la trajectoire, il engendre un cylindre généralisé dont le générateur est obtenu par projection circulaire du contour sur un quelconque plan contenant l'axe (la projection circulaire d'un point P sur un plan Pl est donnée par la première intersection avec Pl de l'arc de cercle passant par P, centré sur l'axe de rotation, perpendiculaire à ce dernier et orienté par ce dernier -fig 1.6.b-)

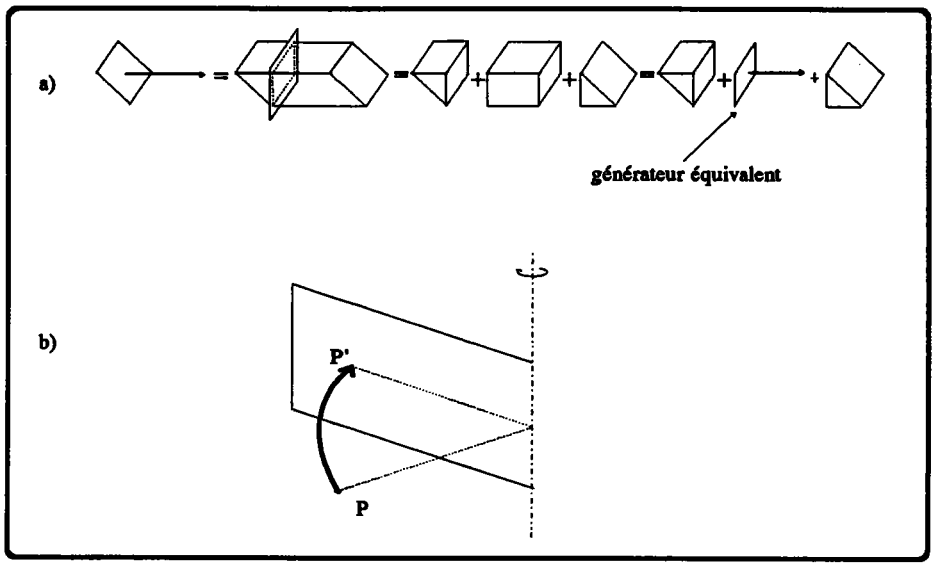


figure 1.6

On ne peut cependant pas étendre ceci au cas de trajectoires quelconques: il est donc faux, contrairement à ce que suggèrent les deux exemples ci-dessus, de généraliser en disant qu'un objet *O*, engendré par un générateur *G* plan et oblique, peut aussi être engendré par un générateur perpendiculaire obtenu par projection curviligne de *G* sur un plan orthogonal à la trajectoire, même aux extrémités près.

La figure 1.7 en donne un exemple.

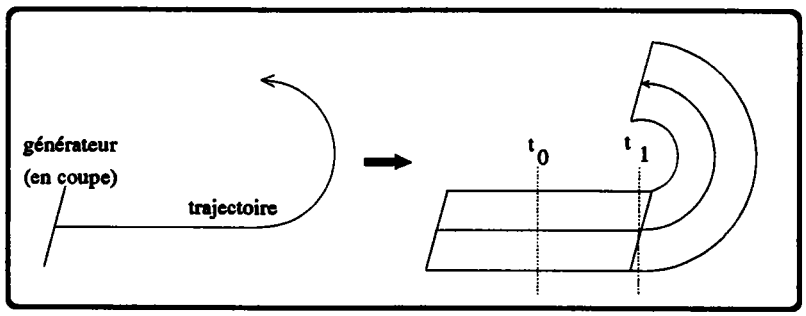


figure 1.7: (vue de côté) les coupes par des plans orthogonaux à la trajectoire en t_0 et t_1 ne sont pas constantes: le volume n'est donc pas un cylindre généralisé.

Les objets engendrés par des générateurs obliques ne sont donc pas, sauf exceptions, des cylindres généralisés.

Pour conclure sur les extrusions d'un générateur plan, nous disons encore un mot de la famille des objets dont le générateur conserve une orientation constante vis-à-vis du repère de référence: à la différence des constructions précédentes, la forme de l'objet engendré ne dépend plus du point auquel on affecte la trajectoire. En contrepartie, ces déplacements sont les plus susceptibles d'engendrer des volumes non valides; en effet, dès que la

trajectoire s'inscrit dans un plan parallèle à celui du générateur, la zone balayée devient localement plane et ne peut donc plus participer à un volume valide (fig 1.8.a).

Ce type de problème ne se pose pas avec les constructions pour lesquelles l'orientation du générateur est fixe par rapport à la trajectoire (et non parallèle); le seul risque est alors que cette dernière soit telle qu'elle induise des singularités comme par exemple un générateur "re-touchant" en un point ou une ligne une zone déjà balayée (fig 1.8.b).

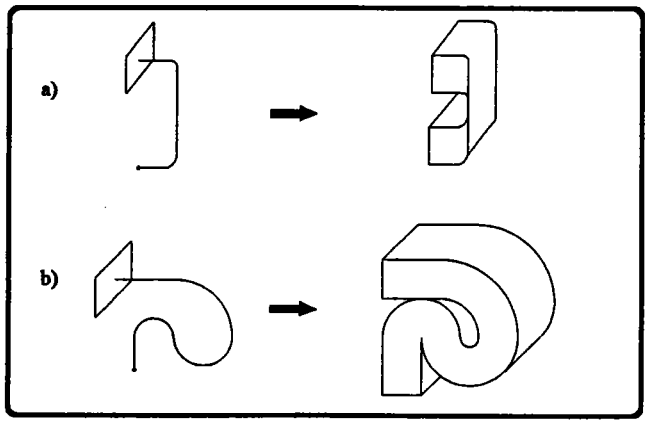


figure 1.8

Le principe consistant à spécifier un déplacement en donnant la trajectoire de l'origine d'un repère local, dans lequel est exprimé le générateur, ne s'applique pas qu'aux générateurs de type contour: on peut de la même façon définir le mouvement d'une ligne ou d'un solide. Nous n'en parlons pas ici puisque cela n'apporte rien au niveau du déplacement.

Dans ce qui précède, nous avons plusieurs fois parlé de trajectoires; voici donc très brièvement quelques moyens d'en définir:

- le plus facile à mettre en oeuvre est de fournir directement la (ou les) équations qui la caractérisent. C'est aussi le moyen se prétant le moins à des manipulations interactives et exigeant le plus de travail de la part de l'opérateur. Ce cas peut cependant survenir sans charge supplémentaire pour l'opérateur lorsque la trajectoire est le résultat d'un calcul (c'est par exemple le cas en commande numérique ou la trajectoire de l'outil est une offset d'une surface gauche).
- préciser des points de passage et/ou des contraintes en ces points. Cette technique est en particulier très utilisée pour les courbes paramétriques cubiques que l'on définit complètement en donnant quatre points de passage ou deux points et deux tangentes etc.

Notons que ceci inclut la possibilité, très utilisée en extrusion, de définir le déplacement d'un générateur en donnant des positions initiale et finale et un mode de raccordement (rectiligne, courbe...).

- par intersection d'un plan et d'une surface (pour une courbe plane) ou de deux surfaces (pour une courbe quelconque).
- par la donnée d'un axe, dans le cas précis d'une trajectoire circulaire
- par une technique d'offset qui consiste à donner une courbe de référence et à en construire une deuxième à une distance *d* donnée (il s'agit là d'une généralisation de la notion de parallèle) [COQ 87a] [HOS 85].

Evoquons enfin les divers moyens d'incorporer une déformation au mouvement. Le plus simple est d'utiliser un changement d'échelle, fonction du même paramètre que la trajectoire, qui amènera le générateur à se dilater ou à se compresser pendant son mouvement.

On peut aussi affecter à plusieurs points du générateur des trajectoires différentes et imposer que ce dernier se déforme de façon à rester en contact avec chacune d'elles pendant son mouvement. Eventuellement en complément de cette technique, on peut faire en sorte que les trajectoires soient parcourues à des vitesses différentes.

La déformation peut enfin être exprimée d'une manière plus implicite, en ne donnant plus une (ou des) trajectoire(s), mais un certain nombre de sections de passage, orientées librement et d'aspects variables: un cône, par exemple, est la donnée d'un cercle et d'un contour réduit à un point. On s'approche alors des techniques de raccordement.

1.2.2. Exemples

1.2.2.1. Déplacement à orientation constante et sans déformation

Nous avons vu qu'un moyen de décrire le mouvement à orientation constante et sans déformation d'un contour 2D est d'exprimer ce dernier dans un repère local et de fixer la trajectoire de l'origine de ce repère (dit **mouvant**). L'orientation constante se traduit par l'invariance de l'angle formé par la normale au plan du contour et la tangente à la courbe. Dans le cas d'un cylindre généralisé où le contour reste perpendiculaire à la trajectoire, on simplifie cette technique en faisant en sorte que le premier vecteur du repère soit confondu avec la tangente; les deux autres forment alors nécessairement une base du plan portant le contour.

Klok [KLO 86] propose un repère mouvant satisfaisant à ces principes: il a la particularité de n'être défini qu'à partir de la trajectoire donc d'être indépendant de sa position dans l'espace, du système de coordonnées et du paramétrage. Il s'appuie sur le repère de Frénet que l'on définit ici sur une courbe paramétrique $C(s)$ (pour s variant dans $[0,L]$) sur laquelle sont faites les hypothèses suivantes:

- $C(s)$ est deux fois continûment différentiable.
- $C'(s)$ est non nul pour tout s de $[0,L]$.
- C est paramétré par l'abscisse curviligne, c'est-à-dire qu'en plus d'être non nulle, la dérivée est constamment de norme 1. Autrement dit, la courbe est parcourue "à vitesse constante" ($||C'(s)|| = 1$ sur $[0,L]$).

En tout point s, les trois vecteurs $t(s)$, $n(s)$ et $b(s)$ du repère orthonormé de Frénet sont alors définis de la manière suivante:

$t(s) = C'(s)$: vecteur tangent unitaire

$n(s) = C''(s) / ||C''(s)||$: vecteur normal unitaire (si $||C''(s)|| \neq 0$)

$b(s) = t(s) \wedge n(s)$: vecteur binormal unitaire ($\wedge$ désigne le produit vectoriel).

Le contour générateur G étant perpendiculaire à la trajectoire, ses équations paramétriques ne dépendent que de la normale et de la binormale, soit, pour une valeur s_0 de l'abscisse curviligne:

$$G(u) = g_1(u).n(s_0) + g_2(u).b(s_0)$$

ce qui amène une équation de la surface cylindrique balayée SP, simplement en déplaçant le repère le long de $C(s)$:

$$SP(s,u) = C(s) + g_1(u).n(s) + g_2(u).b(s)$$

Dans le cas où la trajectoire est ouverte, il suffit de fermer cette surface à l'aide de deux faces planes bordées respectivement par $SP(0,u)$ et $SP(L,u)$ pour obtenir un cylindre généralisé. Dans le cas contraire, le repère de Frénet en $s=0$ se superpose exactement à celui défini en $s=L$: la surface se ferme donc automatiquement et forme un objet topologiquement équivalent à un tore si la trajectoire n'est pas trop torturée. Dans le cas contraire, il se peut que le tuyau engendré fasse des noeuds (fig 1.9.a) ou, si le rayon de courbure est partout suffisamment petit en regard du contour, que l'objet engendré soit topologiquement équivalent à une sphère (fig 1.9.b).

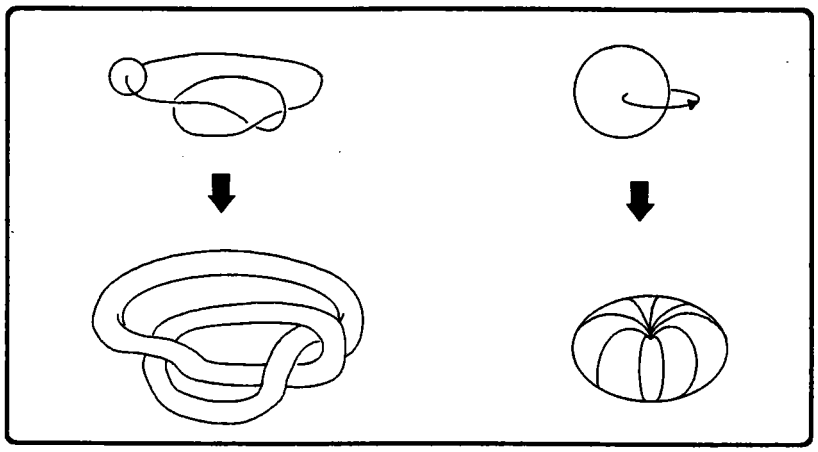


figure 1.9

Nous montrons à présent d'une manière informelle comment la rotation instantanée du repère de Frénet est liée à la torsion de la trajectoire:

- on a $b'(s) = (t(s) \wedge n(s))' = t'(s) \wedge n(s) + t(s) \wedge n'(s) = t(s) \wedge n'(s)$ car $t'(s) = 0$
  donc $b'(s)$ est orthogonal à $t(s)$.
- de plus, on a $b(s) \times b'(s) = (1/2).(b(s) \times b(s))' = (1)' = 0$
  donc $b'(s)$ est orthogonal à $b(s)$
  et, par conséquent, étant perpendiculaire à $t(s)$ et à $b(s)$ qui sont eux-mêmes perpendiculaires à $n(s)$, $b'(s)$ est colinéaire à $n(s)$ ou encore:
- $b'(s) = o(s).n(s)$ où, par définition, $o(s)$ désigne la torsion de la courbe.

De là, on tire que $db / ds = o(s).n(s)$ et $db = o(s).n(s).ds$, ce qui peut être interprété ainsi: pour une petite augmentation ds du paramètre, la binormale est modifiée d'une petite quantité vectorielle $k.n(s)$ ce qui va la déplacer dans le plan perpendiculaire à $t(s)$ dans un mouvement qui peut être assimilé à une rotation dont l'amplitude est fonction de la torsion $o(s)$. Dans le même temps, $n(s)$, restant perpendiculaire à $b(s)$, décrit un mouvement identique (fig 1.10): ceci se traduit par une rotation instantanée des axes orientés par n et b, autour de l'axe orienté par t.

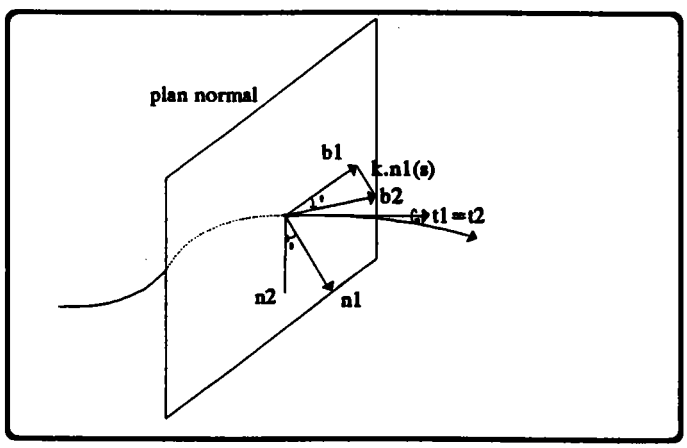


figure 1.10

Tel qu'il a été défini, le repère de Frénet souffre malgré tout de deux inconvénients d'importance:

- le vecteur normal *n(s)* n'est défini que si la quantité $||n(s)||$, autrement dit la courbure, n'est pas nulle: ceci constitue un handicap, en particulier pour les courbes comprenant des portions rectilignes ou les courbes planes comprenant un point d'inflexion; dans ce dernier cas, le repère a en plus le défaut de s'inverser au passage du point d'inflexion (car *n(s)* est toujours orienté vers le centre de courbure) ce qui provoque une discontinuité du mouvement du générateur.
  Comme il est rarissime qu'une courbe non plane présente une courbure nulle, Klok ne se préoccupe que des courbes planes et propose de substituer en tout point à *b(s)* un vecteur constant, unitaire: la normale au plan du contour. En conséquence, pour que le repère reste orthonormé direct, *n(s)* est défini par produit vectoriel de *b(s)* et *t(s)*, ce qui donne un nouveau repère local, continu et défini sur la totalité d'une courbe plane (fig 1.12), en particulier aux points d'inflexion où il ne s'inverse plus (la normale reste toujours d'un même coté de la courbe).
- la deuxième difficulté vient de l'hypothèse d'un paramétrage curviligne: il est vrai que si une courbe est C^1 et régulière (sa tangente n'est jamais nulle) alors il est possible de déduire du paramétrage en place un paramétrage curviligne. Mais si cela se réalise mathématiquement assez facilement (la longueur est le résultat d'une intégrale), il n'en est rien d'un point de vue informatique.

Il faut alors appliquer des formules un peu moins simples mais valables quel que soit le paramétrage (fig 1.11), dès que les hypothèses de régularité et de continuité de la dérivée sont vérifiées [HIL 52]:

$$t(s) = C'(s) / ||C'(s)||$$
$$n(s) = b(s) \wedge t(s)$$
$$b(s) = (C'(s) \wedge C''(s)) / ||C'(s) \wedge C''(s)||$$

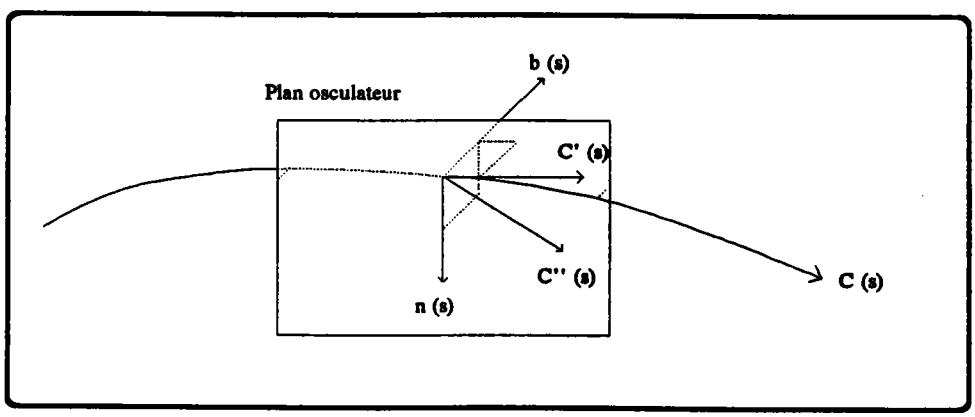


figure 1.11

Ceci dit, le problème de l'annulation de la deuxième dérivée se pose encore et l'on aboutit finalement, en appliquant la solution proposée ci-dessus, à:

$t(s) = C'(s) / ||C'(s)||$

$n(s) = b(s) \wedge t(s)$

$b(s) =$ m, normale au plan de la courbe, si celle-ci est plane

ou

$(C'(s) \wedge C''(s)) / ||C'(s) \wedge C''(s)||$ sinon

A de rares exceptions près, cette dernière formulation du repère de Frénet est applicable à tout type de courbe paramétrique C^1 et régulière. Pour qu'en plus le mouvement du repère soit continu, ce qui paraît indispensable pour une extrusion, il faut une continuité de la deuxième dérivée.

Or il arrive, lorsqu'une courbe est constituée d'un certain nombre de segments (par exemple des courbes de Riesenfeld ou B-splines), que la condition C^2 ne soit pas réalisée aux points de jonction: si, comme c'est généralement le cas, elle est au moins C^1 en ces points, les repères à l'extrémité d'un segment et à l'origine du suivant, au lieu d'être exactement superposés, ne seront égaux qu'à une rotation près autour de leur axe commun orienté par la tangente.

Si par contre la condition C^2 est réalisée, le même problème se pose à nouveau lorsque certains des segments sont plans car la binormale est alors choisie de façon arbitraire: une conséquence de ceci est que, sauf exceptions, une trajectoire n'engendre pas le volume intuitif attendu quand elle comporte des portions planes (fig 1.12).

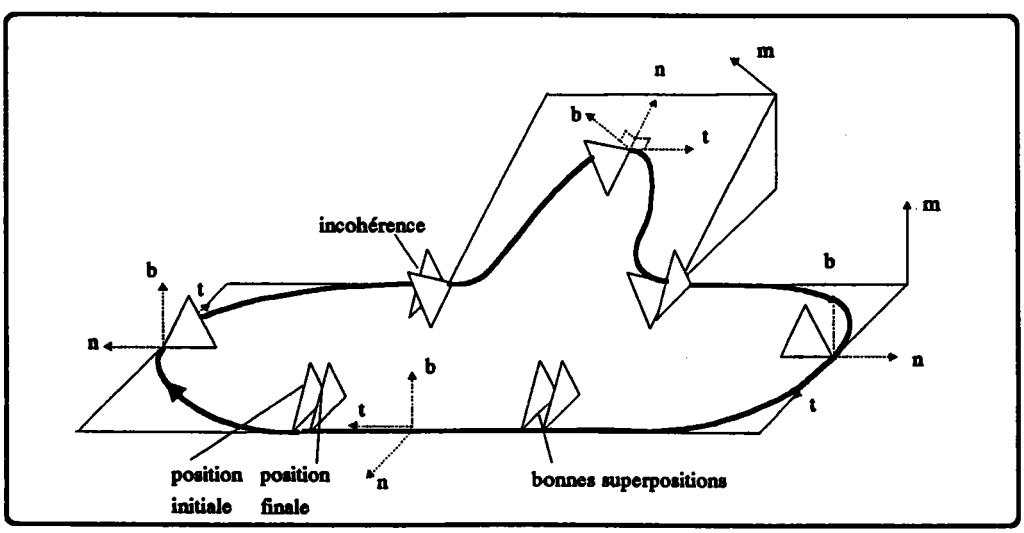


figure 1.12

1.2.2.2. Déplacement à orientation variable

Nous illustrons le déplacement à orientation variable à l'aide de la technique proposée par Lossing et Eshleman [LOS 74] et connue sous le nom de courbe position-direction (PD curves).

Comme précédemment, on donne la trajectoire *t(u)* d'un point. Sur cette trajectoire, on définit un repère mouvant dans lequel est exprimé le générateur, mais cette fois l'orientation du repère varie vis-à-vis de la tangente à la trajectoire.

La variation est décrite par une deuxième fonction *d(u)* qui à tout *u* (le paramètre même de la trajectoire) associe un vecteur direction: le repère mouvant est alors construit à partir de *t'(u)* et *d(u)* de la manière suivante:

$. e1 = t'(u) / ||t'(u)||$

$. e2 = (d(u) \wedge t'(u)) / ||d(u) \wedge t'(u)||$

$. e3 = e1 \wedge e2$

Une variation de d(u) se traduit ainsi par une modification de l'orientation du repère (e1,e2,e3).

Exemple:

Une courbe plane n'a pas de torsion mais la technique de Lossing et Eshleman permet d'en introduire une artificielle: considérons un générateur se déplaçant perpendiculairement à une trajectoire rectiligne autour de laquelle on souhaite le voir tourner de manière à engendrer un tube en spirale (fig 1.13).

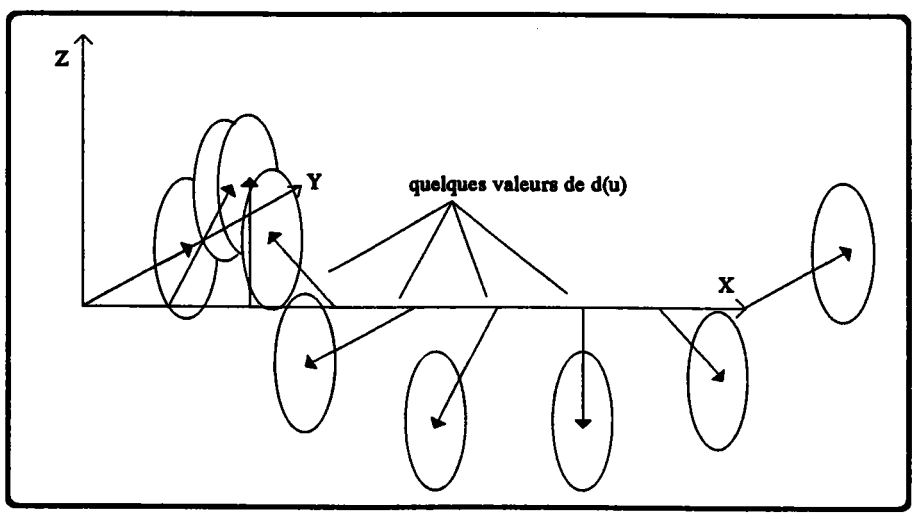


figure 1.13

On définit pour cela un vecteur d(u) constamment orthogonal à t(u) mais subissant une rotation autour de O:

t (u) = (u,0,0) $0 <= u <= 1$

d (u) = (0,cos (2.pi.u),sin (2.pi.u)) $0 <= u <= 1$

donc t'(u) = (1,0,0)

et e1(u) = (1,0,0)
e2(u) = (0,-sin (2.pi.u),cos (2.pi.u))
e3 (u) = (0,-cos (2.pi.u),-sin (2.pi.u))

ce qui définit bien un repère mouvant dont le premier vecteur reste colinéaire à i et dont les deux autres subissent un mouvement continu de rotation: si le générateur est fonction de ces deux vecteurs, il subira effectivement un mouvement de spirale, d'axe (Ox).

1.2.2.3. déplacement avec déformation

Les formes extrudées les plus riches s'obtiennent en combinant position, orientation et déformation. Ce n'est que ce type de déplacement qui permet de concevoir en une seule opération un tuyau de raccord entre deux conduites de diamètres différents, le fuselage d'un avion ou un tube cathodique. Pour illustrer la variété des moyens de quantifier la déformation, nous exposons trois techniques: deux datent déjà d'un certain nombre d'années, la troisième est récente.

La première de ces méthodes est une adaptation immédiate des courbes position-déplacement: elle consiste à faire subir au générateur une homothétie dont le ou les rapports sont fonction du temps. Si un même rapport est appliqué aux trois axes, il y a seulement grossissement ou rétrécissement du générateur; dans le cas contraire, il y a effectivement déformation.

Pratiquement, un moyen de réaliser cette déformation est de choisir un point particulier C du repère mouvant comme centre de l'homothétie et de compléter les fonctions *position* et *direction* d'une troisième fonction *facteur d'échelle* qui en tout point de la trajectoire donne les rapports d'homothétie sur chaque axe du repère mouvant.

Stanley Mathews propose dès 1969 une formulation plus implicite de la déformation en donnant une série de sections de passage du générateur, toutes perpendiculaires à un certain

axe. On construit alors une surface en tendant "une peau" sur ces sections à l'aide d'une technique de raccordement (interpolation, surface réglée - cf 2.4.3. -...). Pour une meilleure description historique, on pourra consulter [ELL 89].

Nous citons enfin Choi et Lee [CHO 90] qui ont récemment étendu cette technique en exprimant mathématiquement quatre types de surfaces extrudées:

a) une généralisation de la surface réglée: une ligne (non restreinte à un segment) se déplace en maintenant ses deux extrémités sur deux courbes, ce qui peut l'amener à se déformer. Pour définir complètement le mouvement, on impose aux deux extrémités de se déplacer à des vitesses proportionnellement égales, c'est-à-dire qu'à tout moment, elles ont parcouru la même proportion de leurs trajectoires respectives.
b) une surface obtenue en déplaçant perpendiculairement à son plan une section qui se déforme de manière à se superposer à des sections de passage d'aspects variables et, simultanément, à rester en contact avec des courbes directrices.
c) une surface engendrée par une ligne subissant un mouvement de rotation autour d'un axe mais se déformant de manière à ce que ses extrémités restent en contact avec deux courbes particulières.
d) une surface engendrée par une section se déplaçant de manière à se superposer à des sections de passage et dont la trajectoire, entre deux sections consécutives, est fixée par une courbe: il s'agit d'une généralisation de b.

1.3. Définition formelle de l'extrusion

Nous avons caractérisé un objet construit par extrusion comme l'ensemble des points balayés par un générateur G dans son déplacement. C'est la définition proposée unanimement dans la littérature, à la formulation près:

- c'est aussi l'ensemble des trajectoires des points de G
- ou, plus formellement:

Soit un générateur G,
soit T l'ensemble des transformations géométriques de $\mathbf{R}^3$ dans $\mathbf{R}^3$,
soit D : [a,b] ------> T
t ------> D (t)
l'application "déplacement" qui, à tout instant t de [a,b] associe la transformation qui amène G sur G_t, le générateur à l'instant t,

Alors l'extrusion de G selon le déplacement D est constituée de l'ensemble des points P de $\mathbf{R}^3$ tels qu'il existe un t dans [a,b] pour lequel P appartient à $D(t)(G) = G_t$.

Nous observons sur quelques cas particuliers les résultats qu'amène cette définition lorsque le générateur est un volume creux. Nous pourrions faire les mêmes constatations sur une ligne discontinue ou une surface trouée mais le déplacement d'un volume a davantage d'applications.

Considérons d'abord un usinage par un outil que l'on suppose être creux. Une application stricte de la définition donne que le volume usiné, obtenu par extrusion du volume outil, est plein si l'amplitude du mouvement est supérieure aux dimensions de la cavité, creux sinon (fig 1.14).

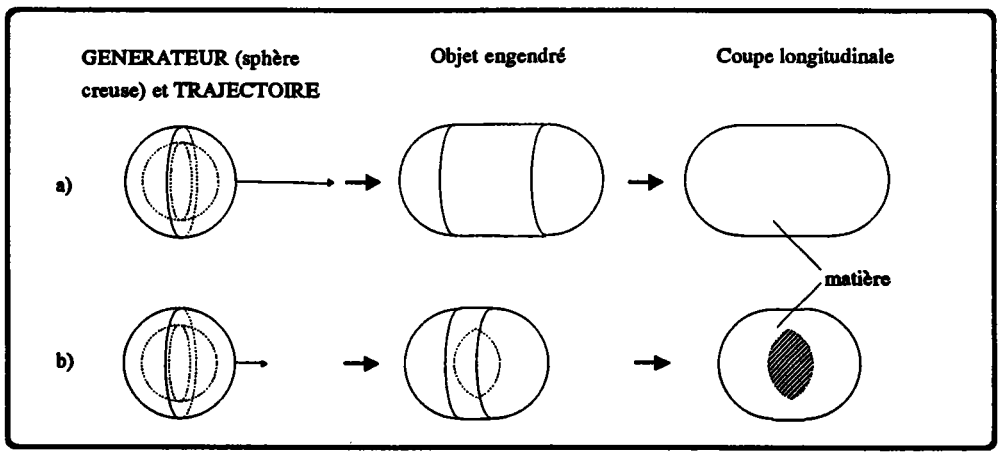


figure 1.14: application de la définition à un générateur creux

A priori, le deuxième résultat peut paraître discutable puisque le volume effectivement usiné est forcément plein. Pratiquement, ce résultat est cependant tout-à-fait acceptable puisque l'outil partant nécessairement d'une zone vide (sans matière), la présence d'une poche interne n'introduira pas d'erreurs.

Cette remarque et d'autres, semblables, que l'on pourrait faire sur l'étude des collisions d'une pièce en mouvement ou de la zone atteignable pour un bras articulé, renforcent donc la définition communément admise.

Cette discussion est mentionnée ici parce qu'il nous semblait plus en accord avec l'intuition que le déplacement d'un volume creux engendre un autre volume creux (fig 1.15), ce qui va à l'encontre de l'actuelle définition. Mais nous n'avons finalement trouvé d'autre justification à cette approche qu'un intérêt purement théorique; nous conserverons donc dans la suite la définition traditionnelle de l'extrusion.

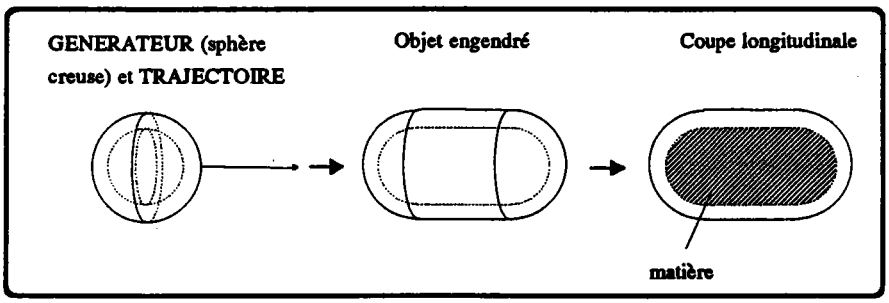


figure 1.15: application intuitive de la définition

1.4. Introduction aux prochains chapitres

Nous avons brièvement, dans ce chapitre d'introduction, donné une définition formelle de l'extrusion et décrit quelques moyens pratiques de spécifier ses paramètres caractéristiques, générateur et déplacement. La logique veut donc que l'on s'intéresse à présent au moyen de modéliser effectivement les objets ainsi définis.

En conséquence, nous proposons dans le prochain chapitre une étude théorique des techniques employées pour stocker des objets extrudés dans les deux grands modes de représentation, par les limites et par arbre de construction.

Nous décrivons ensuite dans une phase plus pratique la mise en oeuvre de deux algorithmes d'extrusion et quelques uns des problèmes sous-jacents soulevés par cette implantation.

La dernière partie prend enfin le contre-pied de ce qui précède et aborde l'extrusion sous un angle différent, celui de l'usinage: nous avons en effet remarqué que les méthodes d'extraction de caractéristiques de forme dans des modèles issus de la conception isolent

souvent des caractéristiques que l'on peut assimiler à des enlèvements de matière donc à des extrusions puisque ces excédents sont appelés à être usinés par un outil en mouvement.

Cette constatation a motivé une étude des techniques d'extraction et quelques réflexions qui font l'objet du quatrième chapitre.

CHAPITRE II

2. MODELISATION D'UN OBJET EXTRUDE

2.1. Introduction

Quand on étudie les différents types de représentation actuellement proposés, on cite en général les approches B-rep, C.S.G. (dont les techniques de décomposition de l'espace), mathématiques, paramétriques et, au même niveau, l'extrusion.

En pratique, il devient de plus en plus difficile de dissocier ces différentes méthodes.

Pour l'extrusion, en particulier, on s'aperçoit, et c'est ce que l'on s'attachera à montrer dans ce qui suit, qu'elle touche de près aux techniques mathématiques et paramétriques, et que si un couple (générateur,déplacement) constitue effectivement un modèle à lui seul, on ne peut pour autant s'en satisfaire, d'où le besoin d'une représentation classique par les limites et/ou par arbre de construction. Nous n'évoquerons pas l'approche par les oct-trees très peu présente dans la bibliographie [BRU 90].

2.2. L'extrusion en tant que représentation

Il apparaît rapidement que l'on ne peut utiliser l'extrusion comme représentation unique d'un système de modélisation géométique car il paraît impossible de réaliser *pratiquement* la fermeture d'une telle représentation, c'est-à-dire qu'il n'est pas toujours possible de mettre en évidence un générateur et un déplacement définissant le résultat d'une combinaison d'objets extrudés. Nous illustrons ceci sur un exemple.

Considérons le calcul de l'intersection régularisée de deux objets: s'ils sont tous deux représentés par un arbre C.S.G., il est possible (très simplement) de stocker le résultat de l'opération dans une structure du même type; il en va de même, dans une certaine limite, pour toute autre opération: *le schéma C.S.G. présente donc*, par rapport à un certain ensemble d'opérations régularisées, *une relative fermeture*.

La même propriété est vérifiée par un modèle B-rep: on sait que l'intersection des deux objets est entourée par une certaine peau, dont le calcul est certes moins simple (surtout s'il prend en compte les surfaces gauches) que ne l'est celui de l'intersection de deux objets C.S.G.: *la représentation par les limites réalise* donc elle aussi cette nécessaire *fermeture*, même si cette propriété est algorithmiquement plus difficile à mettre en oeuvre.

Considérons enfin une opération similaire sur deux objets définis chacun par un couple (générateur,déplacement). Il suffit de cet exemple pour se rendre compte que si, là encore, la fermeture théorique peut être établie pour n'importe quelle opération, il paraît impossible de la réaliser *pratiquement*, c'est-à-dire de trouver, pour le résultat de l'intersection et, plus généralement, pour tout objet O, un couple (Gen,Depl) tel que Extr (Gen,Depl) = O (fig 2.1).

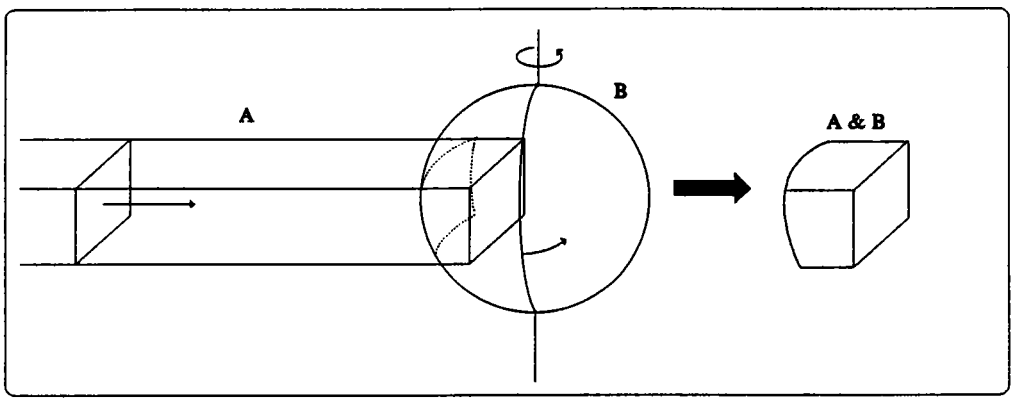


figure 2.1: l'intersection de deux objets extrudés est difficile à exprimer sous forme d'une extrusion.

Cette remarque nous amène à étudier les moyens de représenter une pièce extrudée dans les deux grands modes désormais classiques: l'arbre de construction et l'évaluation des frontières.

2.3. Modélisation d'une extrusion dans un arbre C.S.G.

Les opérateurs ensemblistes sont présents à deux niveaux dans un système de CAO: ils constituent d'abord un puissant outil de conception, permettant de décrire d'une manière concise et intuitive des formes compliquées.

Mais ils sont également une technique de représentation très utilisée aujourd'hui. C'est ce deuxième aspect qui nous intéressera, dans la mesure où nous allons étudier dans quelles limites il peut être appliqué au stockage d'une extrusion.

2.3.1. La non-unicité de la représentation C.S.G.

Le modèle C.S.G. est généralement un arbre binaire. On y trouve, aux feuilles, des volumes primitifs appartenant à une panoplie plus ou moins vaste et aux noeuds non terminaux les trois opérateurs ensemblistes régularisés d'union, d'intersection et de

différence, souvent complétés d'un opérateur de transformation géométrique. On décrit ainsi des objets d'une manière compacte et valide, dans la limite où les volumes primitifs le sont eux-mêmes et où les opérateurs sont régularisés.

Il faut noter, bien que les primitives évoquées ci-dessus soient des volumes, que ce schéma s'applique aussi bien à des objets plans.

Une caractéristique majeure de ce type de représentation est sa non-unicité, qui se manifeste à deux niveaux:

- les propriétés des relations ensemblistes font qu'une même expression peut être formulée de diverses façons:
  - (a U b) U c = a U (b U c)
  - (a - b) - c= a - (b U c) d'où plusieurs arbres équivalents.
- d'un point de vue plus géométrique, on peut créer une même pièce à l'aide de plusieurs jeux différents de volumes primitifs.

Cette non-unicité constitue à la fois un avantage et un inconvénient:

Le fait de jouer avec les différentes formulations d'un même objet et d'exploiter la plus adaptée à un problème donné n'est pas nouveau en soi. C'est selon ce principe que l'on conserve des équations implicite et paramétrique d'une même surface, ou que l'on fait voisiner une représentation exacte du type C.S.G. avec une autre, approchée, du type B-rep. De même, il y a, théoriquement au moins, des avantages à tirer des diverses représentations C.S.G. possibles d'un même objet. Parmi ceux là, citons:

- les possibilités offertes aux méthodes d'optimisation des techniques de lancer de rayons. Il s'agit par exemple de la recherche des volumes redondants ou inutiles [WOO 88a] dont l'élimination implique une diminution du nombre des intersections d'un rayon avec les feuilles de l'arbre (fig 2.2). En combinant ces subtilités logicielles avec d'autres plus matérielles (en attribuant en parallèle un processeur à chaque feuille de l'arbre, par exemple), on arrive alors à des temps d'affichage avec rendu réaliste qui sont presque du temps réel (entre une et six secondes) [GOL 89].

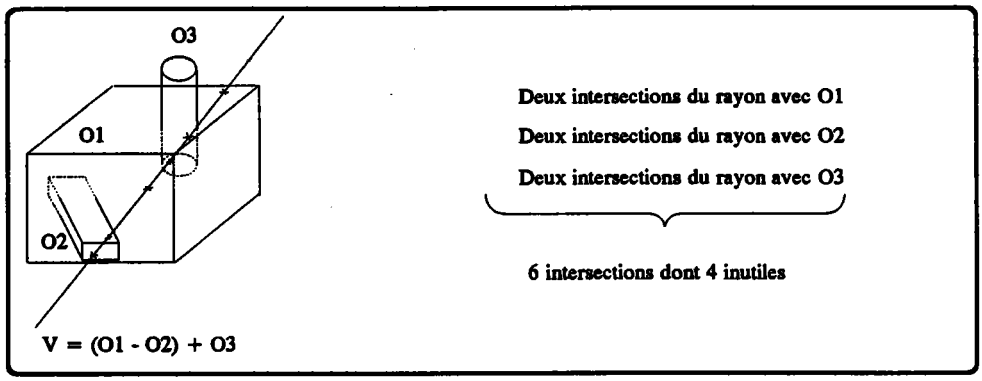


figure 2.2

- dans un contexte d'affichage uniquement, la possibilité de restructurer l'arbre de manière à mieux profiter des englobants, donc, ici encore, à diminuer le nombre d'intersections significatives (fig 2.3).

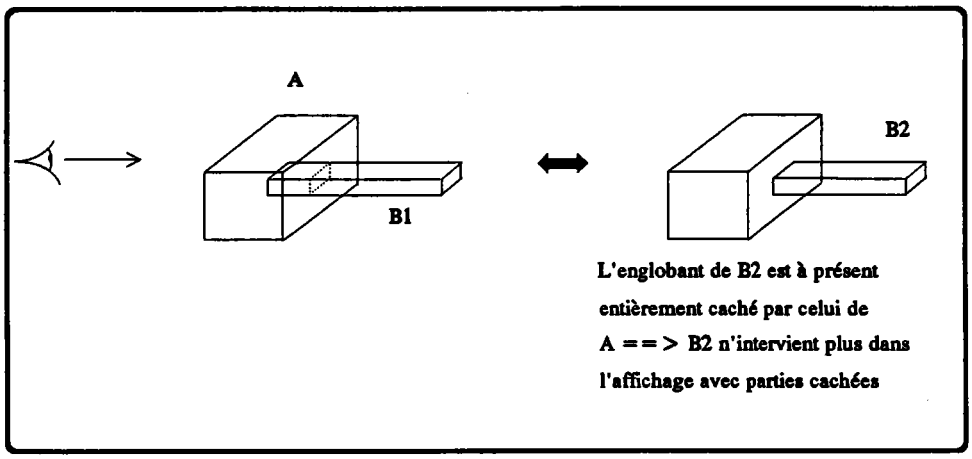


figure 2.3

- au moment de la préparation de la fabrication, la faculté d'exprimer l'arbre C.S.G. sous une forme D.S.G. (Destructive Solid Geometry), dont on déduit plus facilement un brut et la succession des enlèvements de matière aboutissant à la pièce désirée (fig 2.4).

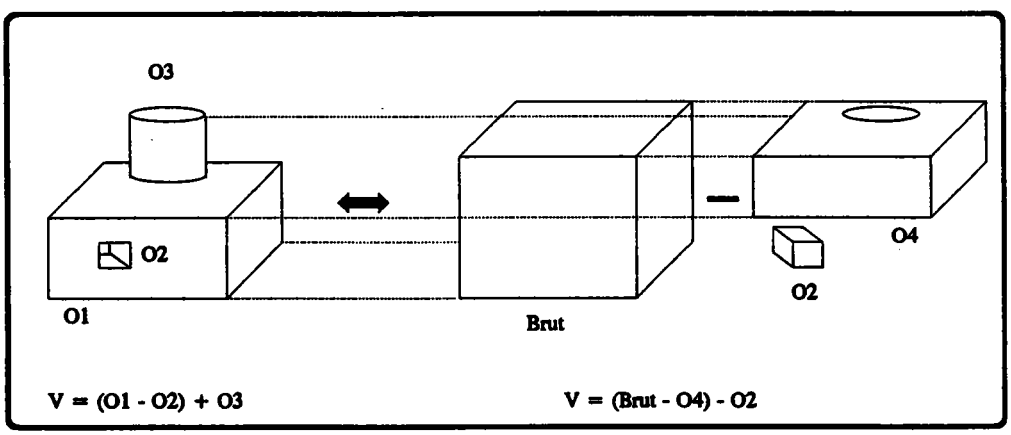


figure 2.4

Il y a cependant une marge entre mettre l'avantage d'une méthode en évidence et l'exploiter effectivement. Reprenons l'exemple des surfaces dont on conserve deux définitions; l'avantage d'une telle stratégie est évident puisque l'on profite des meilleurs côtés (discutés dans [GAR 90]) de chacune d'elles. Malheureusement, personne ne sait encore, dans le cas général, déduire efficacement l'une des formulations de l'autre, ce qui rend impossible le maintien de leur cohérence; or ce problème se pose fatalement à un moment ou à un autre, par exemple lors de la modification interactive de la forme de la surface. Il n'y a donc pas moyen ici de tirer parti d'un avantage qui ne reste que théorique.

Le maintien de plusieurs définitions pose aussi un problème trivial d'occupation de place, non négligeable sur de petites machines.

Pour résumer, nous dirons qu'il y a beaucoup à tirer de la présence simultanée de plusieurs définitions, par exemple les différentes expressions ensemblistes d'un même objet, dans la limite où l'on sait :

- déduire l'une de l'autre,
- choisir la plus adaptée à une situation donnée,
- assurer leur cohérence.

A l'inverse et un peu pour les mêmes raisons, cette possibilité d'avoir plusieurs descriptions ensemblistes équivalentes devient un handicap dans divers domaines: ça l'est quand il s'agit de comparer deux objets dont on veut savoir s'il sont égaux, par exemple. Dans le même ordre d'idée, la non-unicité complique la détection et l'extraction de caractéristiques de formes (features): la première parce qu'une foule de combinaisons peuvent produire la même caractéristique, la deuxième parce que les primitives participant au feature identifié peuvent être réparties dans tout l'arbre: ceci fait que les algorithmes du domaine ne s'appliquent encore qu'à des arbres très particuliers (cf chapitre 4).

2.3.2 Intégration de l'extrusion dans le schéma C.S.G.

L'extrusion partage avec le schéma C.S.G. les propriétés de compacité et de non-unicité, ce qui amène à penser qu'il est possible d'intégrer harmonieusement la première au deuxième. On étendrait ainsi considérablement la puissance de la représentation par arbre de construction, par exemple en tuyauterie ou en cinématique (volume balayé par un objet en mouvement) (fig 2.5).

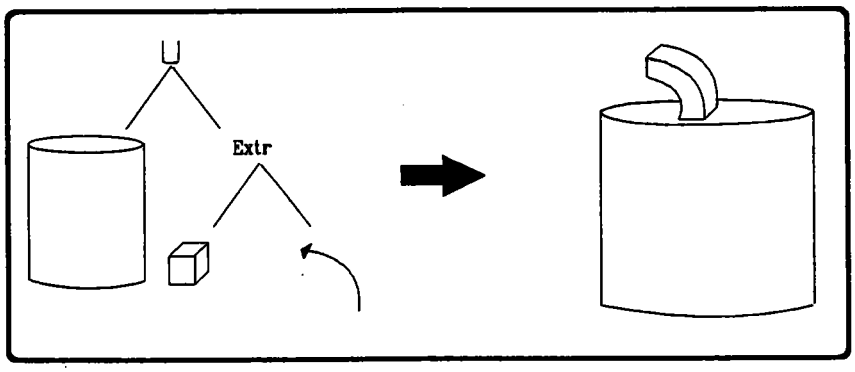


figure 2.5

Pour réaliser complètement cette intégration, il faut prévoir:

- d'étendre le jeu des volumes primitifs aux objets extrudés; on peut remarquer que les volumes primitifs traditionnels peuvent tous être exprimés sous cette forme, qui apparaît donc comme un moyen d'uniformiser la représentation des feuilles de l'arbre.
- la possibilité d'incorporer à un niveau quelconque de l'arborescence l'opérateur "balayage", dont le fils gauche pourrait être le générateur et le droit le déplacement.

Cette intégration soulève des difficultés nouvelles dans tous les algorithmes associés au schéma C.S.G., en particulier dans les méthodes d'affichage direct, avec des procédés du type ray-casting.

Mais elle remet également en cause la validité de cette représentation et divers algorithmes qui s'y appliquent bien que n'utilisant pas de lancers de rayons. Ceci concerne entre autres les techniques de restructuration, visant à isoler des parties caractéristiques (features) ou à diminuer, dans une certaine mesure, la non-unicité de la représentation.

Par ailleurs, nous avons évoqué ci-dessus la possibilité de représenter tous les volumes primitifs à l'aide d'un même schéma, en l'occurrence un couple (*générateur,déplacement*). Il serait intéressant de discuter les apports et les limitations d'une telle approche. Tous ces points sont abordés ou développés dans les paragraphes suivants.

2.3.3. Problèmes posés par l'intégration au niveau des feuilles

Utilisé pour l'affichage (parties cachées, ombrage, réflexion, transparence...), pour l'évaluation de propriétés physiques (volume, masse), le calcul des intersections d'un rayon avec un volume primitif est une opération fondamentale sur un arbre de construction.

Sa complexité dépend étroitement du *nombre* et du *type* des représentations utilisées pour décrire les volumes primitifs; ces dernières dépendent elles-mêmes des fonctionalités souhaitées pour le logiciel.

Concernant le nombre des représentations, trois choix sont possibles:

- une représentation par type de primitive, choisie en fonction de son adaptation à divers calculs, dont l'intersection avec un rayon ou la mise à l'échelle. Il faut ici prendre représentation au sens large: par '*une représentation*', on entend en fait toutes les représentations jugées utiles à la description du volume et de ses propriétés.
  Cette façon de faire permet d'utiliser au mieux les caractéristiques de chaque primitive donc, entre autres, d'améliorer les temps de réponse.
- une unique représentation, unifiée, pour toutes les primitives: outre le fait qu'un unique algorithme d'intersection rayon/objet est alors suffisant, notons que cette approche permet d'étendre facilement, par exemple interactivement, la panoplie des volumes de base; il suffit pour cela que l'objet candidat soit exprimé dans la représentation choisie pour les feuilles ou qu'une conversion soit possible. Une illustration en est donnée dans [SAH 90]: chaque primitive est définie par ses limites, les faces étant modélisées par des B-splines non uniformes (Nurbs).
- la troisième approche est un panachage des deux précédentes: chaque primitive a son modèle particulier mais existent également une ou plusieurs représentations standards et autant d'algorithmes d'intersection rayon/objet associés; ceci permet d'intégrer de nouveaux objets de base dans la limite où ils peuvent être exprimés dans un de ces formats.

Tout comme le nombre de représentations, la nature de chacune d'elles est fonction des opérations que l'on souhaite possibles sur les feuilles. Parmi les opérations les plus fréquemment utilisées, on trouve, pour le citer encore une fois, le calcul des intersections avec un rayon mais aussi la mise à l'échelle (agrandissement ou réduction) et le positionnement dans l'espace. On peut citer encore le changement d'échelle dans une direction donnée, la déformation locale etc.

Deux questions se posent donc: "*Quel modèle utiliser pour décrire une feuille définie par un générateur et un déplacement ? cette définition implicite est-elle suffisante ?*" et "*l'extrusion apporte-t-elle quelque chose de nouveau dans le choix d'avoir une ou plusieurs représentations pour les feuilles d'un arbre C.S.G. ? en particulier, serait-elle une représentation unifiée pratique ?*".

Nous tentons de répondre à ces questions dans les deux paragraphes suivants.

2.3.3.1. Quelle représentation pour une feuille correspondant à une primitive extrudée ?

Trois possibilités s'offrent à nous de représenter une feuille correspondant à une primitive extrudée:

- ne conserver qu'une référence au générateur et au déplacement.
- en trouver une description équivalente sous forme ensembliste, en s'appuyant sur les primitives habituelles: cette technique, peu courante dans la littérature, est abordée en 2.3.6.
- procéder de manière traditionnelle en évaluant ses limites et en considérant l'objet évalué comme feuille de l'arbre. Notons au passage que c'est un moyen d'uniformiser la représentation d'un certain nombre de primitives, en particulier de celles construites par balayage de l'espace, mais que cela suppose la mise en oeuvre d'algorithmes d'évaluation des frontières ce qui va à l'encontre de la vocation de concision du schéma C.S.G. et amène à conserver deux représentations redondantes. C'est donc une solution de dépannage qu'il y aurait intérêt à éviter; nous verrons toutefois dans la suite que c'est parfois le seul moyen de faire. Par conséquent, les méthodes de calcul des limites exposées en 2.4. pourront être exploitées dans le cadre d'une approche du type arbre de construction.

Avant de faire un choix, il devient nécessaire de préciser quels types d'objets extrudés doivent figurer parmi les volumes de base.

Supposons que l'on tolère aux feuilles des objets à deux dimensions comme les plans ou les surfaces (c'est-à-dire dont les générateurs ont une ou quelquefois deux dimensions); a priori, étant donné que l'on prévoit d'intégrer aussi un *opérateur* d'extrusion, ce ne devrait pas être un facteur bloquant pour la définition de volumes. Mais cela implique que les opérateurs ensemblistes peuvent être appliqués à des objets non volumiques et que la

propriété selon laquelle tout sous-arbre décrit un volume valide n'est plus vérifiée. A supposer que pour remédier à cela on fasse en sorte que seul un opérateur d'extrusion puisse être appliqué à une feuille 2D, on ne peut toujours pas garantir la validité de l'objet engendré: il se peut, par exemple, qu'il n'ait localement aucune épaisseur.

Pour ces raisons, il est préférable que les primitives extrudées soient, comme les autres, des volumes.

Précisons alors la nature de leur générateur: il paraît inutile de prévoir le cas où ce dernier est lui-même un volume: on empiète alors sur le domaine de l'*opérateur* d'extrusion proprement dit (cf 2.3.4.). Il reste donc les générateurs à une (points, lignes) et deux (plans, surfaces) dimensions; si la deuxième catégorie reste le moyen le plus aisé d'engendrer des volumes, il n'y a, a priori, pas de raison d'interdire un générateur uni-dimensionnel. Considérons par exemple un segment suivant une trajectoire fermée: on définit facilement un volume à l'aide de la surface réglée qu'il engendre et des deux faces bordées par les trajectoires de ses extrémités.

Les générateurs à une ou deux dimensions ont malgré tout le handicap d'engendrer des volumes dont on teste difficilement la validité. Il y a pourtant des cas où une vérification est facile ou du moins réalisable:

- face plane et translation non parallèle au plan de la face (prisme),
- face plane et trajectoire circulaire autour d'un axe situé dans le plan de la face et ne coupant pas cette dernière (volume topologiquement équivalent à un tore),
- ligne plane en rotation autour d'un axe qu'elle touche en ses extrémités seulement (volume topologiquement équivalent à une sphère),
- face plane et trajectoire dont la tangente conserve, par rapport à la face, une orientation constante et non parallèle au plan de la face,

mais d'une manière générale, l'orientation et la forme du générateur pouvant varier, il est extrêmement difficile de vérifier la validité des objets de base extrudés. Ceci dit, le même problème se pose dans des approches plus traditionnelles, comme celle qui consiste à décrire une primitive comme l'intersection de plusieurs demi-espaces.

Nous raisonnerons donc dans la suite sur des volumes primitifs extrudés supposés valides et engendrés par des générateurs à une ou deux dimensions.

Considérons alors le problème fondamental dans l'approche C.S.G. du calcul des intersections d'un rayon avec une primitive définie implicitement par un couple (*gen,depl*).

La difficulté vient évidemment de ce que la frontière du volume balayé n'est pas connue: on n'en a ni une définition mathématique, explicite (z = f (x,y) par exemple), implicite (f(x,y,z) = 0) ou paramétrique (x=f(u,v), y=g(u,v), z=h(u,v)), ni une caractérisation approchée, par des facettes par exemple.

Nous voyons les réponses apportées à ces difficultés par quatre auteurs: Kajiya [KAJ 83], Van Wijk [VAN 84] et Bronsvoort et Klok [BRO 85].

a) *méthode de Kajiya [KAJ 83]*

Kajiya propose des méthodes pour le calcul des intersections d'un rayon avec un prisme droit ou une surface de révolution radiale.

Dans les deux cas, il explicite aussi le calcul de la normale aux points d'intersection car elle est indispensable aux algorithmes de rendu mais nous n'évoquerons pas ce point.

Le principe, pour sa méthode et celles exposées ensuite, est de ramener le calcul des intersections d'un rayon et d'une surface (prismatique ou de révolution) à celui des intersections d'un transformé du rayon avec une courbe, donc à l'intersection de deux lignes. Dans ce cas précis, les deux courbes ont en outre l'avantage d'être coplanaires.

Nous commençons par le prisme droit, défini par un contour de base d'équation C(s), un vecteur normé N perpendiculaire au plan de C(s) et une hauteur h (fig 2.6): en quels points le rayon R(t) = A + t U coupe-t-il le prisme droit (C(s),N,h) ?

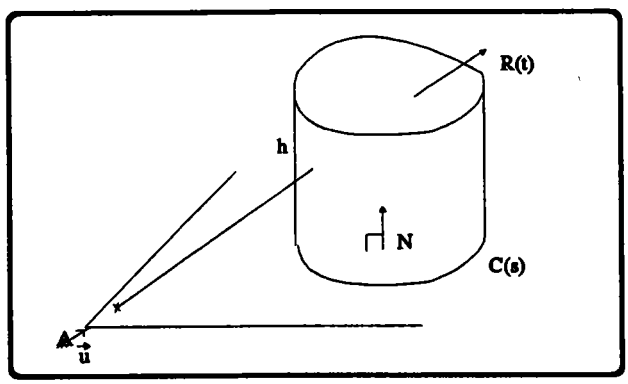


figure 2.6

Le cas le plus simple est celui d'un rayon parallèle au plan de la base: si *R* est à une distance *d* inférieure ou égale à *h* du plan et si, évidemment, *R* est du bon côté du plan, il se peut qu'il coupe le prisme : les éventuelles intersections sont alors celles de *C(s)* avec le projeté du rayon sur le plan, élevées d'une hauteur *d*.

Considérons à présent le cas d'un rayon oblique : soient alors t_0 et t_1 les abscisses sur le segment des intersections de *R(t)* avec le plan de base et son homologue (fig 2.7):

. $I_0 = A + t_0 U$ est une intersection de *R* avec le prisme si I_0 appartient à la portion du plan de base bordée par *C(s)* .

. $I_1 = A + t_1 U$ est une intersection de *R* avec le prisme si le projeté de I_1 sur le plan de base appartient à la zone bordée par *C(s)* .

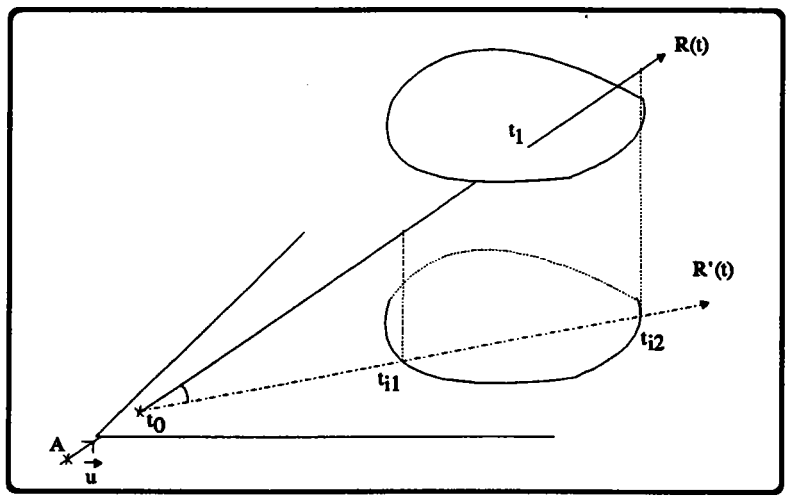


figure 2.7

Ceci nous donne les éventuelles intersections de R avec les deux faces fermant le prisme. Nous déterminons à présent celles avec les faces latérales.

Soit *R'* le projeté de *R* sur le plan de base et t_i les abscisses de ses intersections avec *C(s)*. Un t_i donné correspond à un unique point P_i de *R*, situé à une distance *d* du plan de base. Cette distance est fonction de l'inclinaison de la droite par rapport au plan (fig 2.8):

$$d = | t_i - t_0 | \; U \times N$$

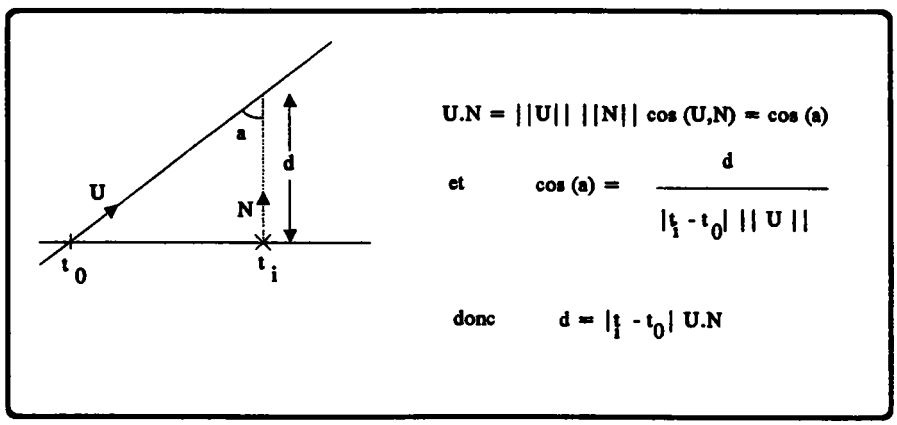


figure 2.8

Finalement, le point P_i ainsi défini est une des intersections recherchées s'il est à une hauteur compatible avec celle du prisme, autrement dit si et seulement si:

$$0 < d < h$$

On constate que l'opération de base est le calcul des intersections du projeté du rayon avec la base du prisme: il importe donc de l'optimiser autant que faire se peut: pour cela, l'auteur utilise la méthode dite des '*strip trees*' consistant à calculer des englobants rectangulaires de portions de plus en plus petites de la courbe de base, que l'on suppose exprimée paramétriquement: on a à la racine de l'arbre un englobant de la courbe entière, immédiatement en dessous les englobants des deux moitiés et ainsi de suite, récursivement (moitié est ici un peu abusif car la courbe n'est pas coupée en son milieu mais en un point de contact avec l'englobant).

L'arbre intervient comme un filtre dans le calcul des intersections: si le rayon ne coupe pas l'englobant général, il n'y a pas de solution. Dans le cas contraire, on cherche ses intersections avec les deux sous-englobants. On progresse ainsi dans l'arbre jusqu'à ne plus trouver d'intersection avec un englobant ou à arriver à une feuille. Suivant la précision désirée, on peut alors se contenter des intersections avec l'englobant (supposées confondues), avec sa diagonale (si on assimile la courbe à un segment de droite) ou avec le segment de courbe.

Le temps perdu à calculer le *strip tree* est amplement justifié par le fait que ce dernier est ensuite réutilisé pour chaque rayon.

Certaines des hypothèses peuvent paraître restrictives mais il n'en est rien:

- la méthode peut être appliquée à une arête oblique: il suffit pour cela de remplacer "projection orthogonale" par "projection oblique" et "élévation" par "translation d'un vecteur parallèle à l'arête". Nous verrons cependant (en d.) qu'il n'est même pas utile de développer ce cas car il trouve une solution plus élégante dans les transformations géométriques.
- nous avons supposé que *C(s)* est exprimé paramétriquement mais seulement pour faciliter le calcul des intersections rayon projeté/courbe de base: il n'est pas interdit d'envisager une formulation implicite qui faciliterait les tests d'appartenance de I_0 et I_1 respectivement à *C(s)* et à son analogue.

  Rien non plus ne va contre le fait que *C(s)* soit une courbe par morceaux.

Nous voyons maintenant comment calculer les intersections d'un rayon $R(t)=A+t.U$ avec une surface de révolution radiale (B,V,ray(s)).

Une surface radiale est la donnée d'un axe $Ax(s)=B+sV$ et, pour tout s, d'un rayon *ray(s)* donnant le rayon du méridien situé à la "hauteur" s (fig 2.9). Ce type de formulation restreint légèrement la notion d'objet de révolution au sens où on l'entend habituellement puisqu'il interdit des objets toriques par exemple.

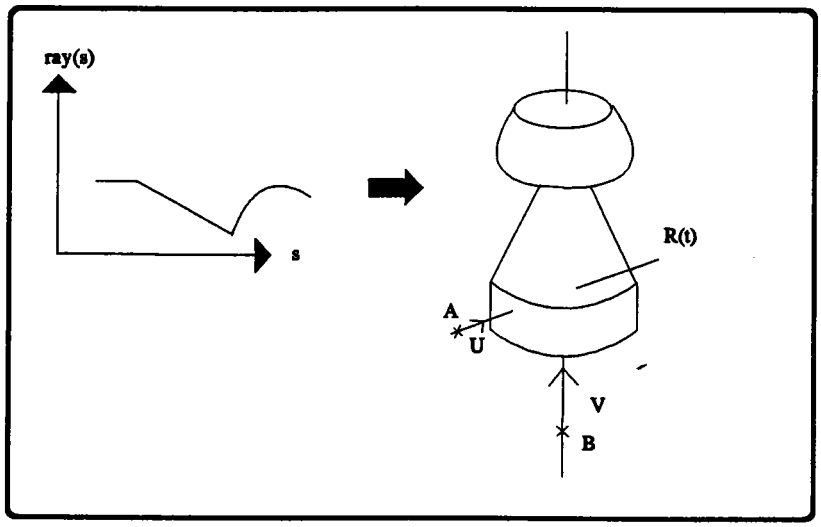


figure 2.9

Dans le même esprit qu'auparavant, l'objectif est ici encore de ramener le calcul des intersections rayon/surface au calcul des intersections du rayon avec deux courbes C_1 et C_2 obtenues en coupant la surface par un plan parallèle à l'axe Ax et contenant le rayon R (fig 2.10).

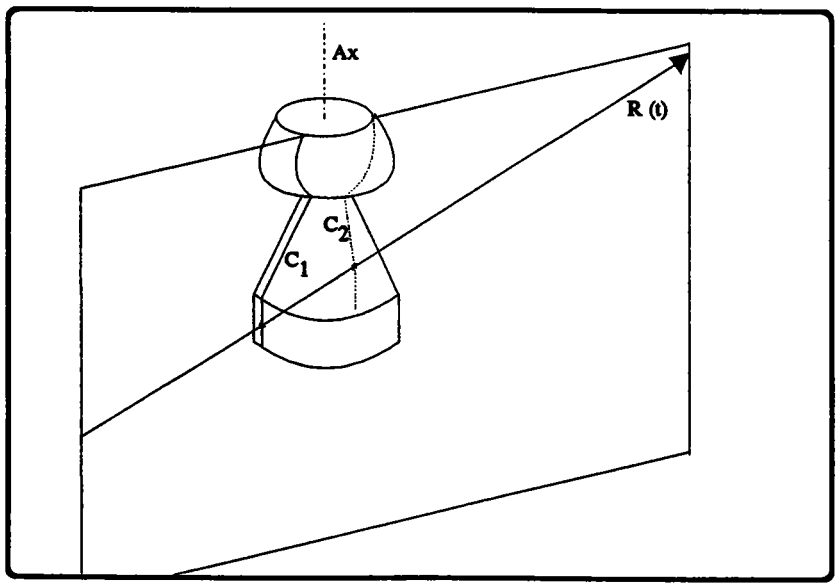


figure 2.10

Exprimons les deux courbes dans un repère, explicité ci-dessous, qui soit lié au plan de coupe (fig 2.11):

. la normale au plan de coupe est $N = V \wedge U$ ($\wedge$ désigne le produit vectoriel)

. l'origine du repère local est le projeté B' de B sur le plan;
notons d la distance qui les sépare, autrement dit la distance de l'axe au plan

. le premier axe du repère est orienté par $W=V \wedge N$, le deuxième par V, le troisième par N.

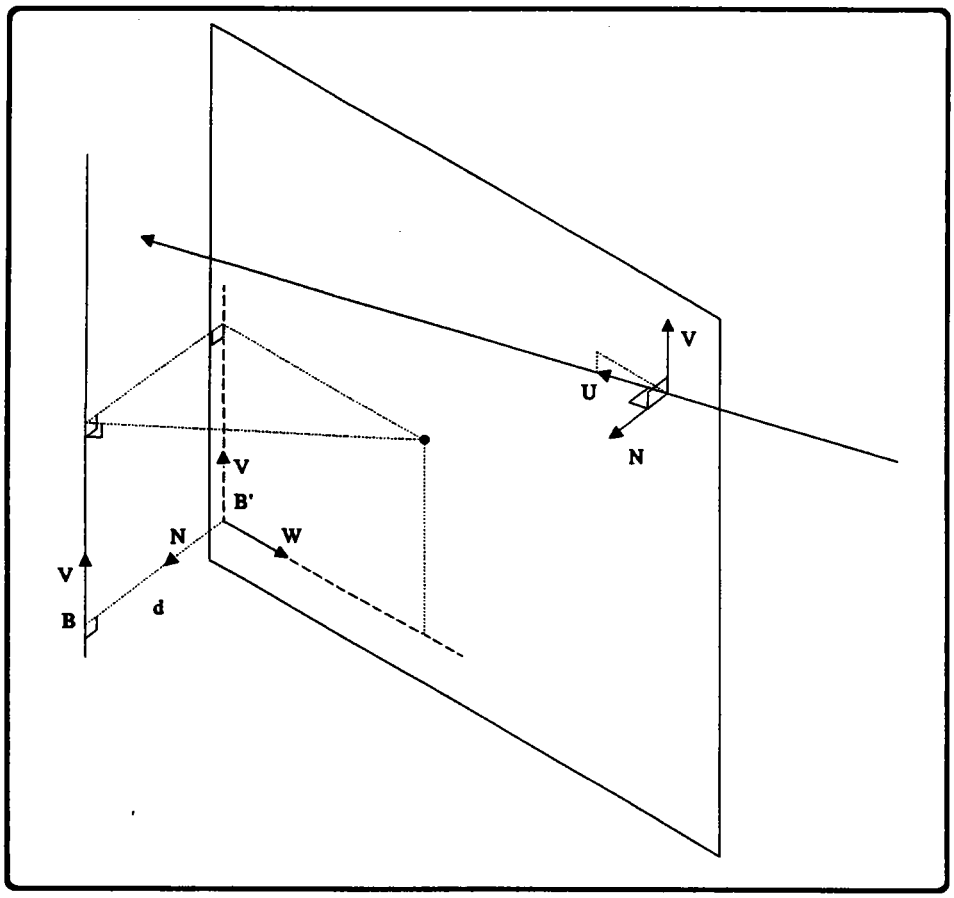


figure 2.11

Un point P de coordonnées (x,s,z) dans le repère local appartient alors:

- au plan si $z=0$
- à la surface si la distance à l'axe est *ray (s)* autrement dit si

$$\sqrt{(x^2+d^2)} = ray(s) \text{ ou encore } x=+- \sqrt{(ray^2(s)-d^2)}$$

Finalement, on en déduit les équations paramétriques de C_1 et C_2 dans le repère local du plan:

$$C_1(s) = (\sqrt{ray^2(s) - d^2}, s)$$

et

$$C_2(s) = (-\sqrt{ray^2(s) - d^2}, s)$$

Quand $ray(s) < d$, c'est-à-dire aux endroits où les méridiens ont un rayon trop petit pour toucher le plan, les courbes C_1 et C_2 n'existent pas (elles ne sont en fait pas définies car la quantité sous la racine est alors négative).

A nouveau, le problème se résume au calcul des intersections du rayon, exprimé lui aussi dans le repère local, avec deux courbes, ces trois objets étant coplanaires.

La méthode telle qu'elle est exposée est assez limitative d'une part à cause de la formulation radiale de la surface, d'autre part parce que les objets décrits ne sont des volumes que si $ray(s)$ s'annule aux bornes de l'intervalle dans lequel s varie et nulle part ailleurs. On peut pourtant envisager de l'étendre sans réelle difficulté:

- dans le cas où une extrémité du méridien ne touche pas l'axe, il suffit pour fermer l'objet de considérer une face plane circulaire perpendiculaire à l'axe et de rayon $ray(s_{extr})$; l'intersection avec cette face est simpliste.
- pour décrire un générateur plus compliqué, en particulier un coutour fermé, il suffit de le subdiviser en plusieurs segments exprimés chacun radialement. On arrive alors à décrire un objet torique par exemple.
  Certains contours resteront cependant inaccessibles comme ceux dont une portion rectiligne est perpendiculaire à l'axe.

  Nous retrouverons cette notion de segmentation du générateur dans les travaux de Van Wijk.

b) *méthode de Van Wijk [VAN 84]*

Van Wijk propose une méthode couvrant le même domaine que celle de Kajiya, les prismes droits et les révolutions, mais introduit aussi un nouveau type d'extrusion, dit "conique". Nous ne citerons donc que les apports pour le prisme et donnerons quelques précisions sur l'opérateur conique, qui a la qualité d'intégrer une déformation d'une manière très simple.

Au niveau du prisme, les deux points à retenir sont:

- l'usage d'un contour défini par morceaux, ce qui permet d'associer un englobant à chacun d'eux et d'optimiser le calcul des intersections avec le rayon projeté R' de la manière suivante (nous supposerons le contour se trouver dans (xOy)):

  SOIT (*xmin,ymin,xmax,ymax*) l'englobant d'un segment de courbe (fig 2.12),
  ALORS
  R' ne coupe pas le morceau si,
  quand sa pente est négative, (*xmin,ymin*) et (*xmax,ymax*) sont d'un même côté de R',
  quand sa pente est positive, (*xmax,ymin*) et (*xmin,ymax*) sont d'un même côté de R'

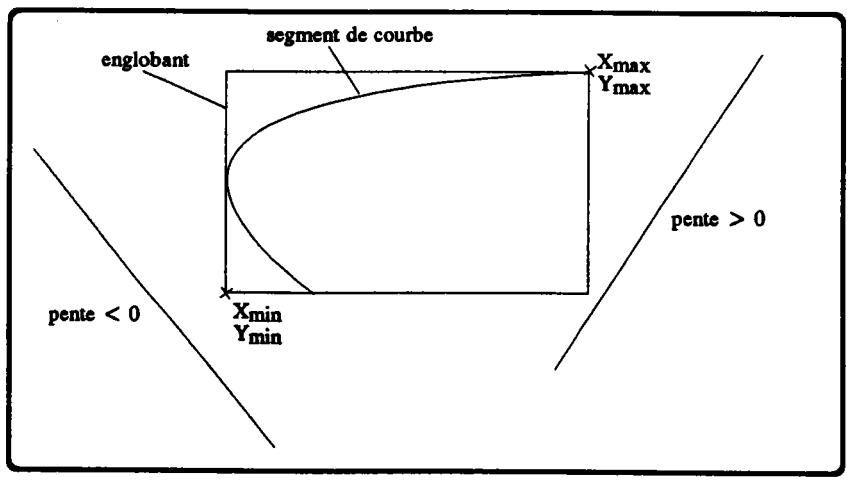


figure 2.12

- une subtilité, basée sur l'algorithme de la demi-droite, évitant de tester l'appartenance aux deux bases des points d'intersection du rayon avec les plans bordant le prisme.
  En reprenant les principes employés au paragraphe précédent, appelons t_i,t_{min} et t_{max} les abcisses des intersections du rayon projeté R' avec le contour et les deux plans (fig 2.13).

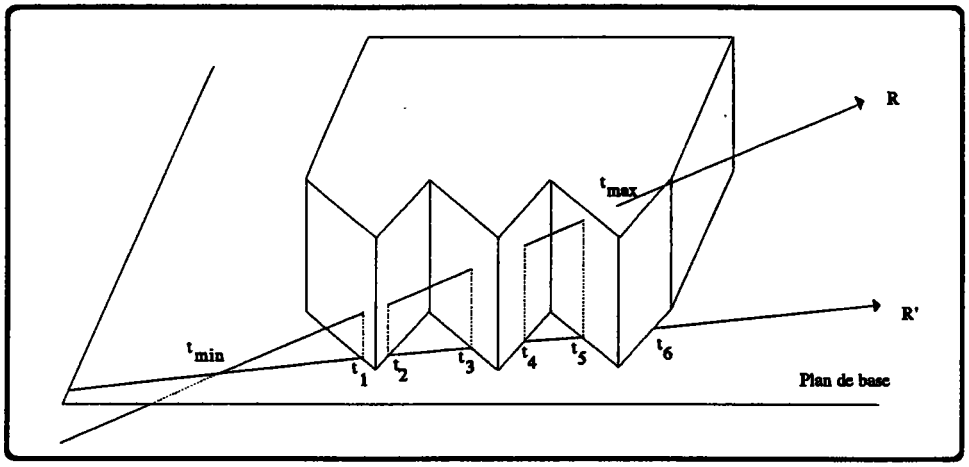


figure 2.13

Cette fois,à la différence d'avec la méthode de Kajiya, on conserve les intersections dont l'abscisse est supérieure à t_{max} (mais pas celles avant t_{min}) bien qu'elles soient a priori inutiles: il suffit alors de compter le nombre d'intersections situées au delà de t_{max} pour savoir si celle correspondant à t_{max} doit être conservée: elle ne doit l'être que si ce nombre est impair. Ceci fait, on retire toutes les abscisses supérieures à t_{max}, éventuellement t_{max} et on ne conserve t_{min} que s'il reste un nombre pair d'intersections.

Nous détaillons à présent le calcul des intersections d'un rayon avec ce que Van Wijk appelle une extrusion conique et qui est en fait un cône dont la base, définie paramétriquement par $(u(s),v(s))$, se trouve dans le plan $z=1$ et le sommet en O (fig 2.14).

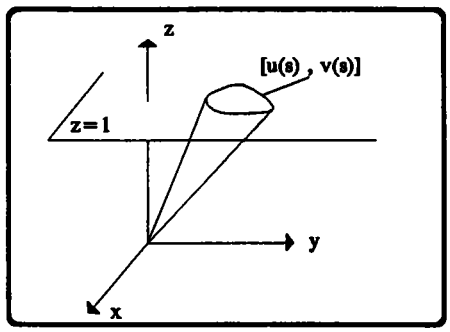


figure 2.14

Un point $P(x,y,z)$ appartient au cône si

- (x,y) appartient à un contour proportionnel à $(u(s),v(s))$, soit :

$$x.v(s) = y.v(s)$$

et

- la proportion entre ce contour et le contour de base est fonction de la hauteur:

$$x = z.u(s)$$

En combinant (1), (2) et les équations paramétriques du rayon, on aboutit à une équation en s, d'un degré inférieur ou égal à trois si $u(s)$ et $v(s)$ sont des cubiques.

Comme précédemment, on évite un calcul de point intérieur au contour $(u(s),v(s),1)$ en comptant les intersections au delà de t_{max} (t_{max} est l'abscisse de l'intersection du rayon avec $z=1$).

c) *méthode de Bronsvoort et Klok [BRO 85]*

Les méthodes de Kajiya et de Van Wijk s'appliquent à des objets extrudés simples, en ce sens que les trajectoires ne peuvent être que droites ou circulaires. Dans ce qui suit, le mouvement est beaucoup plus général puisque la trajectoire est cette fois une courbe presque quelconque de $\mathbf{R}^3$.

De ce fait, la méthode gagne en complexité et mérite quelques explications informelles avant que ne soit détaillé le point de vue mathématique.

L'objectif est de déplacer un contour plan perpendiculairement à une trajectoire $t(u)$ quelconque de l'espace; on souhaite que le contour respecte le comportement local de la trajectoire, à savoir que si celle-ci est animée d'un mouvement de torsion, l'orientation du contour dans son plan doit varier en conséquence (fig 2.15): il s'agit donc des cylindres généralisés introduits en 1.2.

Revenons à présent au problème de l'intersection d'un rayon R avec la surface engendrée par le contour. Considérons une valeur u_0 donnée du paramètre; le contour se trouve alors dans un plan $Pl(u_0)$, passant par $t(u_0)$ et perpendiculaire à la trajectoire. Le rayon coupe ce plan en un point I_{u0}.

Prenons de même un certain nombre d'autres instants $u_1,u_2 \ldots u_i \ldots u_n$, chacun définissant une nouvelle intersection I_i: à chacune de ces intersections correspond de manière naturelle un point I'_i dans le plan $Pl(u_0)$ (fig 2.15). Ce point est obtenu en déplaçant fictivement $Pl(u_i)$ le long de la trajectoire, en le soumettant toujours à la torsion de cette dernière, jusqu'à ce qu'il soit confondu avec $Pl(u_0)$: la position alors occupée par I_i sur $Pl(u_0)$ est le point I'_i.

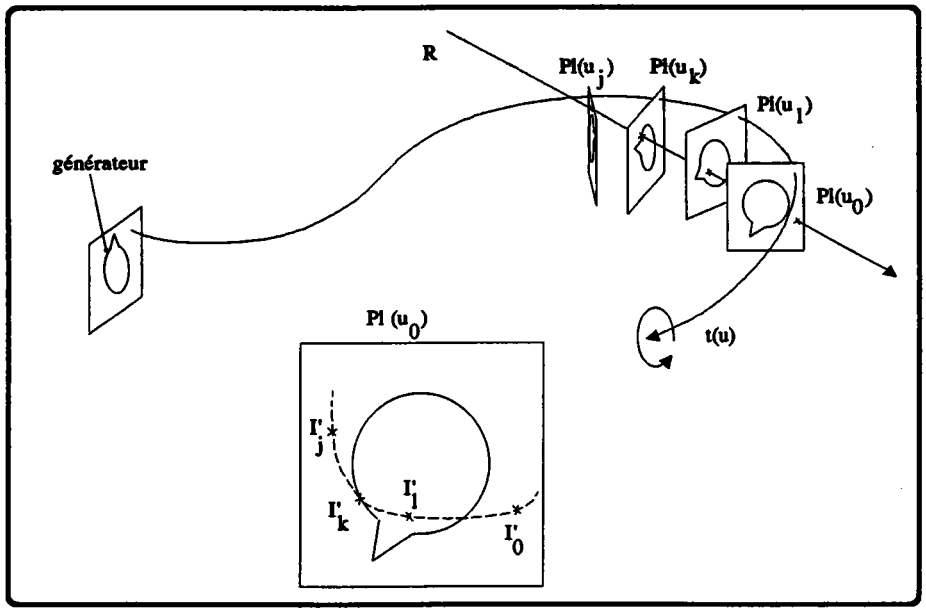


figure 2.15

En faisant varier u entre ses valeurs minimale et maximale, on définit ainsi sur $Pl(u_0)$ une courbe $I'(u)$ caractéristique des intersections du rayon avec tous les plans $Pl(u)$ et, par conséquent, avec les instances successives du contour: autrement dit, les points d'intersection $u_{i1},u_{i2},\ldots,u_{in}$ de $I'(u)$ avec le contour porté par $Pl(u_0)$ correspondent à des intersections du rayon avec les contours portés par $Pl(u_{i1}),Pl(u_{i2}),\ldots,Pl(u_{in})$, c'est-à-dire à autant d'intersections avec le cylindre généralisé.

Nous reprenons à présent le tout par le détail en faisant l'hypothèse que le cylindre généralisé ne se recoupe pas lui-même.

Comme nous l'avons vu en 1.2.1., un bon moyen d'exprimer le mouvement d'un contour plan engendrant un cylindre généralisé est de lier le contour à un repère de Frénet, glissant sur une courbe paramétrique t(u) et construit ainsi:

e1 = t'(u) / ||t'(u)||
e2 = e3 ^ e1
e3 = **si** la courbe est plane **alors** m, la normale au plan
sinon e1 ^ t''(u) / ||e1 ^ t''(u)||.

Sous ces hypothèses, le contour à un instant u donné et la surface balayée par le contour sont respectivement:

$$C(v) = Cx(v).e2(u) + Cy(v).e3(u)$$

et

$$S(u,v) = t(u) + Cx(v).e2(u) + Cy(v).e3(u)$$

Pour trouver la courbe plane I'(u) introduite ci-dessus, il suffit d'exprimer les intersections du rayon R(s) avec les plans Pl(u) dans le repère mouvant (t(u),e1(u),e2(u),e3(u)).

C(v) étant exprimée dans ce même repère, leur intersection donnera les u auxquels le rayon touche un générateur; il suffira ensuite de ramener les points trouvés dans le repère de référence.

Pour cela, nous exprimons d'abord le rayon dans le repère mouvant, autrement dit, nous faisons subir à R(s) = A + s.D le changement de repère amenant du repère de référence au repère local. Celui-ci est constitué dans cet ordre d'une translation ramenant t(u) sur O puis de trois rotations autour des axes Ox, Oy, Oz ramenant (e1,e2,e3) sur (i,j,k).

La transformation amenant (i,j,k) sur (e1,e2,e3) a pour matrice:

$$M = (e1,e2,e3)$$

donc celle que nous cherchons, son inverse, est ${}^tM$. Finalement, le rayon devient:

$$R(s) = {}^tM.[A + s.D - t(u)].$$

Son point d'intersection avec Pl(u) est caractérisé par une composante nulle selon e1 d'où:

$$e1(u) \times [A + s.D - t(u)] = 0 \qquad (\times = \text{produit scalaire})$$

$$\text{ou} \quad t'(u) \times [A + s.D - t(u)] = 0 \qquad \text{puisque } e1(u) = t'(u)/||t'(u)||.$$

Si t'(u) x D n'est pas nul, autrement dit si le rayon n'est pas parallèle à Pl(u), on en déduit que:

$$s = \frac{t'(u) \times [A - t(u)]}{t'(u) \times D} \qquad (1)$$

La courbe plane I'(u) est constituée de l'ensemble des points d'intersection exprimés chacun dans leur repère local, soit

$$I'(u) = {}^t(e2\ e3) \,.\, [A + s.D - t(u)]$$

$$\begin{cases} e_2(u) \times \left[A - t(u) + \dfrac{t'(u) \times [A - t(u)]}{t'(u) \times D} . D \right] \\ e_3(u) \times \left[A - t(u) + \dfrac{t'(u) \times [A - t(u)]}{t'(u) \times D} . D \right] \end{cases}$$

e2(u) et e3(u) étant eux-mêmes fonction de t'(u) et t''(u), on exprime finalement chaque composante de I'(u) comme une polynomiale en u, mais sous une forme relativement compliquée.

Pour calculer finalement les intersections du rayon avec le cylindre généralisé, on effectue successivement les trois opérations suivantes:

- résolution en u de l'équation I'(u) = C(v), pour trouver les intersections du rayon avec I'(u).
  Nous ne détaillons pas ici l'algorithme d'intersection; il s'agit d'une méthode par subdivision car une résolution analytique n'est pas possible.
- calcul des s correspondant avec (1).
- calcul des points réels correspondant avec I = A + s.D.

Comme nous l'avons vu, le plan du contour se déplaçant le long de la trajectoire et étant soumis à sa courbure et sa torsion, il peut arriver qu'il soit parallèle au rayon. Dans un voisinage de telles valeurs de u, la quantité t'(u) x D est alors proche de 0 et I'(u) tend vers l'infini. La procédure d'intersection ci-dessus coupe donc I'(u) en plusieurs segments et traite à part les cas de parallélisme en calculant directement les intersections du rayon avec le contour.

Pour terminer sur cette méthode, disons un dernier mot de l'hypothèse de non-recoupement: elle comprend évidemment le cas d'une trajectoire décrivant une boucle telle que le cylindre finit par pénétrer à nouveau une zone déjà balayée. Mais elle interdit aussi toutes les trajectoires dont la courbure devient suffisamment forte vis-à-vis de la taille et de la position du contour dans le repère local pour faire que des générateurs successifs se coupent (fig 2.16); en effet, une intersection du rayon avec une instance du générateur n'est alors plus significative d'une intersection du rayon avec le cylindre généralisé.

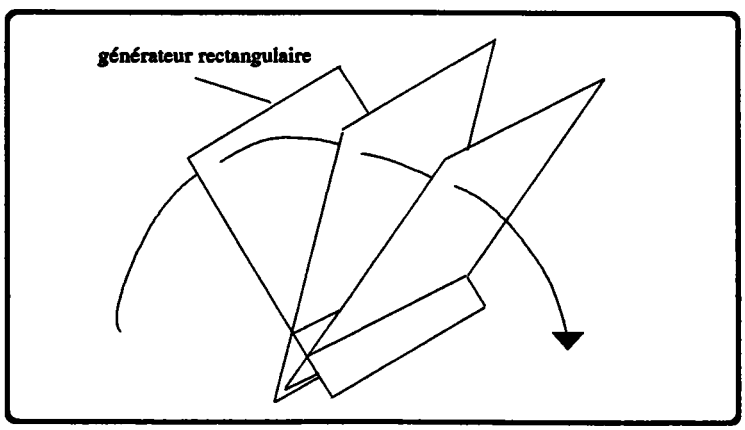


figure 2.16

d) *commentaire*

Les méthodes exposées dans les paragraphes précédents autorisent l'affichage par des procédés de type 'lancer de rayon' d'objets extrudés engendrés par un contour plan et une trajectoire vis-à-vis de laquelle ils conservent une orientation constante. Dans un cas, le déplacement comprend une déformation (prisme conique) mais il semble possible d'intégrer une homothétie aux cylindres généralisés sans trop de difficultés.

Une restriction commune aux deux premières méthodes est d'imposer pour les prismes une arête perpendiculaire à la base. Nous voyons ici qu'il suffit d'une transformation géométrique pour faire d'un prisme droit un prisme oblique et, donc, ouvrir toute la gamme des prismes aux arbres C.S.G.

Soit un prisme $\mathcal{P}$ de base $\mathcal{B}$ et d'arête (a,b,1) (fig 2.17.a). Celui-ci est l'image du prisme normalisé $\mathcal{P}'$ de base $\mathcal{P}$ et d'arête (0,0,1) par la transformation de matrice:

$$\begin{pmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix}$$

donc l'arbre décrivant $\mathcal{P}$ est celui de la figure 2.17.b: on se ramène donc au cas d'un prisme droit.

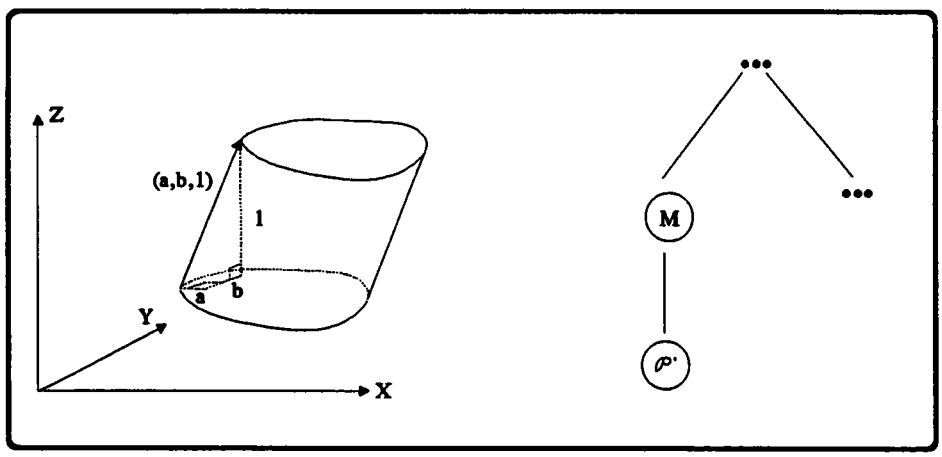


figure 2.17

2.3.3.2. L'extrusion fournit-elle une représentation unifiée pratique des feuilles d'un arbre C.S.G. ?

Nous avons évoqué dans le paragraphe précédent trois possibilités quant au nombre de représentations utilisées pour les feuilles d'un arbre de construction:

- toutes les primitives sont exprimées dans le même schéma.
- 1 schéma par type de primitive.
- une approche mixte.

Nous avions aussi remarqué que les volumes primitifs traditionnels peuvent tous être exprimés sous la forme (*générateur plan,déplacement*) (figure 2.18). Il paraît donc justifié de se demander si cette formulation pourrait être une représentation unifiée des feuilles d'un arbre. Nous reprenons donc les points militant pour ou contre l'unification et les discutons dans le contexte de l'extrusion.

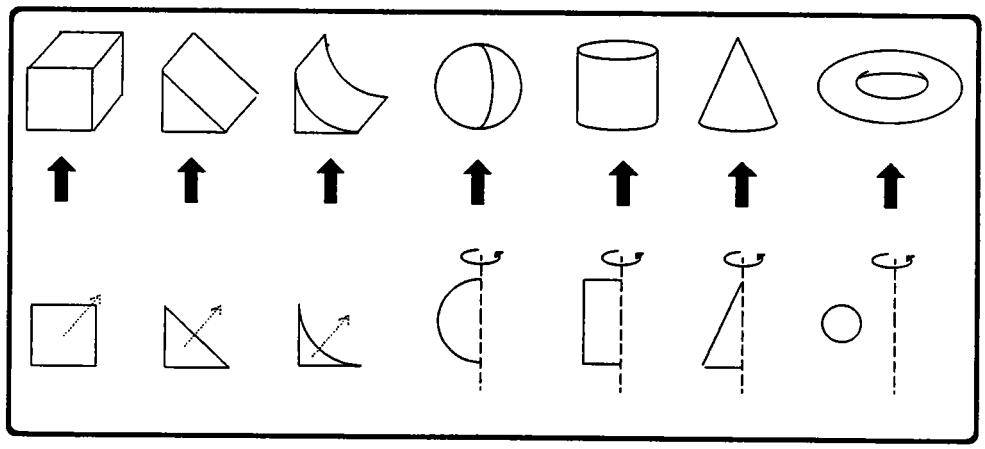


figure 2.18

Arguments en faveur d'une approche unifiée par extrusion d'un générateur plan:

. capacité d'intégrer de nouveaux volumes primitifs: l'extrusion a un haut pouvoir de description permettant d'enrichir considérablement la palette des objets de base et, donc, la portée de la modélisation par arbres C.S.G.: mais elle reste malgré tout insuffisante dans certains domaines, dont la description de polyèdres quelconques par exemple.

. besoin d'un unique algorithme d'intersection rayon/feuille: le fait d'avoir une représentation unique des feuilles autorise à ne programmer qu'une méthode d'intersection mais à condition que l'on sache effectivement le faire dans le cas général: or, d'après ce qui a été vu en 2.3.3.1., le mieux que l'on puisse faire est

d'extruder un contour plan dont le déplacement ne comprenne rien d'autre qu'une trajectoire, sans déformation ni variation de l'orientation du contour par rapport à cette dernière; certes, la majorité des extrusions respectent ces hypothèses mais, ne serait-ce qu'à cause des quelques cas qui s'en détachent, on ne peut employer un couple (*générateur,déplacement*) comme représentation unifiée.

Arguments en faveur d'une approche au cas par cas (une représentation par type de primitive):

. possibilité de concevoir des algorithmes exploitant au mieux les spécificités de chaque type de feuille: chaque primitive a donc sa propre représentation et, en l'occurrence, il est ici possible, pour intégrer des volumes primitifs du type prisme (à base quelconque), objet de révolution (à section quelconque) ou tronc de cone (à base quelconque), d'utiliser les algorithmes ciblés de Kajiya et Van Wijk, plutôt que d'exprimer ces objets sous la forme d'un générateur et d'une trajectoire en vue d'employer la méthode plus générale de Bronsvoort et Klok, qui serait ici moins performante.

Ce qui précède plaide donc finalement en faveur de l'approche mixte, dans laquelle on associe à chaque primitive sa représentation propre et les algorithmes qui vont avec, le tout complété de quelques représentations dites standards, elles-mêmes complétées de leurs bibliothèques de calcul.

Nous préconisons donc:

- pour une sphère, de conserver au moins son centre et son rayon (la forme canonique, donc), dont on déduit facilement des équations algébriques ou paramétriques et d'écrire un algorithme d'intersection exploitant au mieux la symétrie de la sphère.
- des idées similaires pour les tores, cylindres, boîtes, rayons, coin ...
- pour un prisme droit à base quelconque, de conserver le vecteur directeur de l'élévation, une hauteur, un lien avec le contour, que l'on peut supposer construit dans une version 2D du logiciel, et d'implanter l'algorithme de Kajiya.
- pour un prisme oblique d'utiliser la transformation affine vue en 2.3.3.1.d, qui ramène au cas précédent.
- pour un objet de révolution, d'utiliser la méthode de Kajiya en la complétant pour qu'elle s'applique à un solide, ou celle de Van Wijk.

- de définir une représentation standard du type extrusion satisfaisant aux hypothèses de Bronsvoort et Klok.
- de définir une représentation standard pour les polyèdres [VER 87]

...

Le deuxième point de discussion relatif à la description des feuilles d'un arbre portait sur le choix de la représentation, celui-ci dépendant des traitements possibles sur la feuille. Nous en sommes arrivés, pour une primitive extrudée, à conserver la définition implicite (*générateur,déplacement*).

La encore nous n'arrivons pas à une solution idéale: elle est certes convenable car elle permet, dans *certains* cas déjà exposés, le calcul des intersections rayon/primitive ou le positionnement dans l'espace. Elle se prête également à une certaine forme de paramétrage mais d'une manière différente et par certains aspects plus vaste que les autres primitives: ces dernières sont en général paramétrées par quelques grandeurs algébriques (rayon, hauteur, centre ...) qu'il est parfois possible de rattacher à des considérations géométriques mais de façon relativement compliquée (rayon=distance(O_1,O_2), centre=extrémité d'un segment ...). Les objets extrudés sont eux aussi paramétrables à l'aide de grandeurs algébriques (hauteur du prisme, angle de révolution ...) mais la présence du contour (pour être rigoureux, d'une entité géométrique *générateur*) amène d'autres possibilités qui seront évoquées en fin du chapitre 3.

En contrepartie et sauf dans certains cas simples (prismes,objets de révolution) qui n'apportent guère au pouvoir de description d'un arbre C.S.G. traditionnel, on ne sait pas assurer la validité d'un objet engendré par un générateur plan: les feuilles extrudées n'étant pas toujours valides, il en va forcément de même pour l'arbre qui les porte. Il est dommage de perdre une qualité aussi précieuse.

2.3.4. Problèmes posés par un opérateur d'extrusion

Les lignes qui précèdent traitent des possibilités d'intégrer des objets extrudés aux feuilles d'un arbre C.S.G. Nous y avons vu entre autres comment calculer les intersections d'un rayon avec un tel objet. Nous évoquons donc maintenant le problème de la prise en compte d'un opérateur d'extrusion à un noeud non terminal de l'arbre.

Dans un premier temps, considérons l'aspect primordial de l'établissement d'une partition d'un rayon en portions hors, dans et sur une pièce décrite par un arbre de

construction. Avec un arbre habituel, ceci se réalise très simplement en appliquant les opérations booléennes de l'arbre à des portions d'une droite.

Quand intervient un opérateur d'extrusion, le problème se complique considérablement et il semble que personne n'ait encore proposé de solution complète, autrement dit que l'énoncé

"Soit un arbre C.S.G. décrivant un objet O en regard duquel on sait établir une partition d'une quelconque droite D.
L'objet O subit un déplacement, engendrant ainsi un volume V: comment établir une partition d'une quelconque droite D' vis-à-vis de V ?"

reste pour l'heure sans réponse.

On est donc contraint, pour intégrer une telle opération dans un arbre de construction, d'en évaluer les limites et d'en faire une feuille d'un nouvel arbre.

Nous ne proposons dans ce paragraphe que des éléments de réflexion sur le sujet puis citons les résultats de Van Wijk sur l'extrusion d'une sphère.

2.3.4.1. Axes de réflexion

Pour calculer une décomposition d'une droite par rapport au volume VB balayé par un autre volume V lui-même défini par un arbre C.S.G., une première méthode vient en appliquant strictement la définition: on a

$$VB = U\ V_t$$

Il suffit donc théoriquement de décomposer la droite par rapport à tous les V_t (on sait décomposer par rapport à V donc à V_t) puis de faire l'union des partitions obtenues.

Pratiquement, un moyen de réaliser ceci est d'échantillonner et, donc, de ne calculer les intersections qu'avec un nombre fini de V_t. La méthode souffre évidemment des défauts de la discrétisation à savoir:

- il y a risque de manquer des intersections importantes (fig 2.19) et de faire des erreurs non négligeables sur l'affichage ou le calcul du volume.

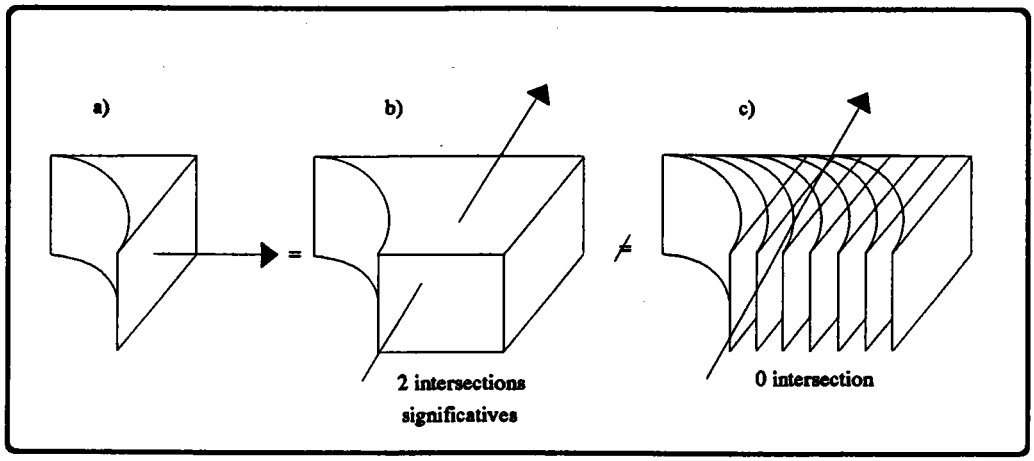


figure 2.19

- il y a augmentation du volume des données, mais pas dans des proportions exorbitantes: en effet, plutôt que de conserver des instances du générateur (donc des sous-arbres), on ne garde que les transformations géométriques T_t qui transforment V en V_t: la décomposition de la droite D vis-à-vis de V_t est alors obtenue en appliquant la transformation T_t à la décomposition d'une transformée de la droite vis-à-vis du générateur V:

$$\text{Decomp}(D,V_t) = T_t(\text{Decomp}(T_t^{-1}(D),V)$$

et, par conséquent

$$\text{Decomp}(D,VB) = U(\text{Decomp}(D,V_t)) \quad , \quad \text{pour t dans } \{t_1,t_2,\ldots,t_n\}$$

Un deuxième moyen de faire consiste à calculer effectivement les intersections de D avec toutes les instances du générateur en mettant une évidence l'équation que doivent vérifier ces intersections. Il s'agit là de la même approche que Bronsvoort et Klok pour les cylindres généralisés mais avec une complication bien supérieure des calculs. Les étapes vers cette équation sont:

- exprimer le volume générateur dans un repère local: sa description est alors constante au fil du mouvement (relativement au repère local).
  La première grosse difficulté est justement là car si l'on veut poursuivre le parallèle avec la méthode de Bronsvoort, l'expression en question doit être une équation du volume, ce qui est quasiment impossible à produire. A priori, deux possibilités s'offrent à nous:
  1) caractériser effectivement le volume en utilisant une formulation paramétrique s'appuyant sur une triple sommation (Box-splines [DOK 90]) ou une formulation implicite (comme c'est possible pour la sphère, par exemple). Mais l'une comme l'autre sont irréalistes du fait de la définition par historique que nous avons du volume.

2) réduire la difficulté en cherchant une équation de la surface de l'objet, autrement dit en évaluant les frontières. Même dans cette hypothèse, il est probable que l'on n'obtiendra pas une mais plusieurs équations, correspondant aux faces de la frontière, ce qui compliquera d'autant les étapes suivantes.

- exprimer la droite dans le repère local.

Ces deux premiers points reviennent en quelque sorte à fixer le générateur et à déplacer D de manière à ce que le mouvement relatif de l'un par rapport à l'autre soit le même que lorsque c'est effectivement le générateur qui bouge et la droite qui est immobile.

- dans le repère local, calculer l'équation caractérisant les intersections de la droite avec le générateur. La encore, il s'agit d'une étape très difficile puisque l'on manipule des équations à quatre ou cinq paramètres.

- traduire l'équation trouvée dans le repère de référence puis dans le repère de la droite.

- ne conserver que les intersections correspondant à des extrema de l'abscisse curviligne: ces points sont les entrées et les sorties du rayon dans/de la matière (fig 2.20).

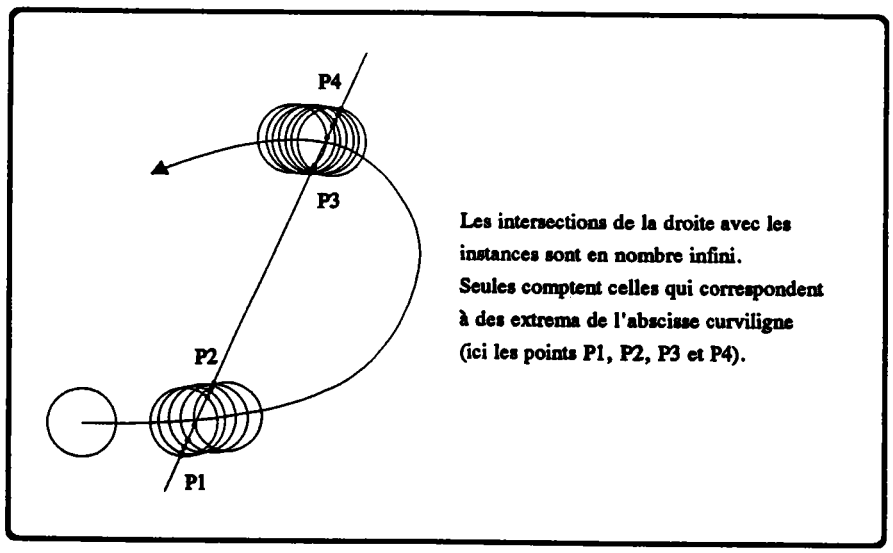


figure 2.20

Cette dernière difficulté ne se posait pas pour les cylindres généralisés car on y a fait l'hypothèse que la surface tubulaire ne se recoupait pas. Ici, le générateur étant un volume, deux instances successives se recoupent forcément.

Il est clair que cette deuxième méthode semble irréaliste tant les difficultés posées sont importantes.

Une dernière approche consiste à exploiter les résultats obtenus sur les cylindres généralisés: sous certaines conditions, dont l'invariance de l'orientation du générateur par rapport à la trajectoire, le volume balayé VB engendré par un volume V est le même que celui engendré par la silhouette de V [MOR 85]. Aux extrémités près, VB est donc un cylindre généralisé: il peut être complètement défini dans un arbre C.S.G. par l'union de ce cylindre et des instances de V aux positions initiale et finale, ce qui autorise aussi bien des opérations d'affichage que de calcul du volume.

2.3.4.2. Exemple: extrusion d'une sphère

Concernant l'extrusion d'un volume, la seule tentative que nous ayons recensée dans la bibliographie en rapport avec la représentation C.S.G. est celle de Van Wijk [VAN 85].

Elle permet le calcul des intersections d'une droite avec le volume balayé par une sphère se déplaçant avec un rayon variable, donc s'applique à des opérations de visualisation mais aussi à des calculs de volumes, de masse... Nous en donnons ici une explication informelle.

Le mouvement de la sphère est paramétré par la trajectoire C(u) de son centre et une valeur variable r(u) du rayon. La droite est exprimée sous la forme $D(t) = A + t.V$.

Observons une sphère se rapprochant de la droite: à l'instant u1, elle la touche pour la première fois en un point $A + t1.V$ (fig 2.21.a); un instant plus tard, l'intersection est, dans la majorité des cas (nous nous contentons d'exposer les principes et n'abordons pas les exceptions), constituée de deux points (fig 2.21.b) et ainsi de suite jusqu'à ce que la sphère soit de nouveau tangente, en u2 (fig 2.21.c).

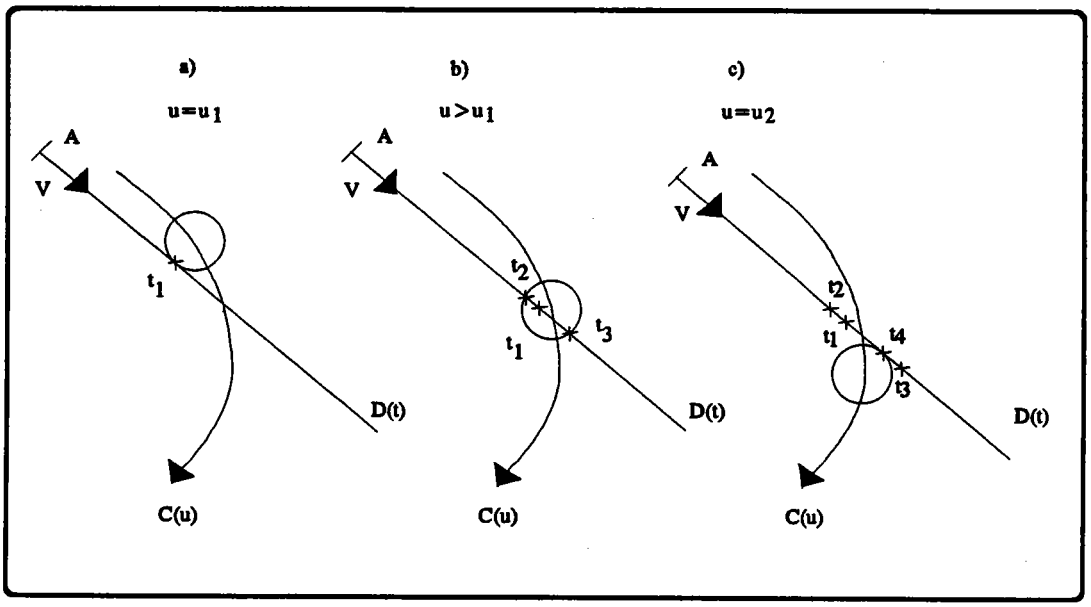


figure 2.21

Pour u variant de u1 à u2, on établit donc une liste des abscisses ti (sur la droite) des intersections droite/sphères, dont seules la première et la dernière nous intéressent.

Dans un premier temps, la méthode consiste donc à exprimer t en fonction de u, en combinant une équation implicite de la famille de sphères ($|P - C(u)|^2 - r(u)^2 = 0$) avec les équations paramétriques de la droite. Les deux t cherchés correspondent à des extrema de t(u) donc sont ceux qui annulent la dérivée t'(u)=dt/du de t(u) (soit t1,t2,t3,t4,t5 sur la figure 2.23).

Il peut arriver, lorsque la trajectoire présente des points anguleux ou lorsque r(u) n'est pas dérivable ou aux extrémités de la trajectoire, que t'(u) n'existe pas. On incorpore alors directement à la liste de points les intersections du rayon avec les instances du générateur en ces instants (t6,t7 sur la figure 2.23).

Toutefois, certains des points calculés dans cette première étape sont encore à éliminer à cause d'un éventuel recouvrement de l'objet engendré ou de cas particuliers introduits par une dérivée non définie.

Si seules des opérations simples d'affichage sont prévues, on contourne la difficulté en ne prenant que la première de ces intersections. Dans le cas contraire, on raisonne ainsi: le volume engendré est constitué d'un certain nombre de volumes tubulaires (des cylindres généralisés en fait), correspondant aux portions dérivables de la trajectoire, et d'un certain nombre de sphères correspondant aux discontinuités (fig 2.22).

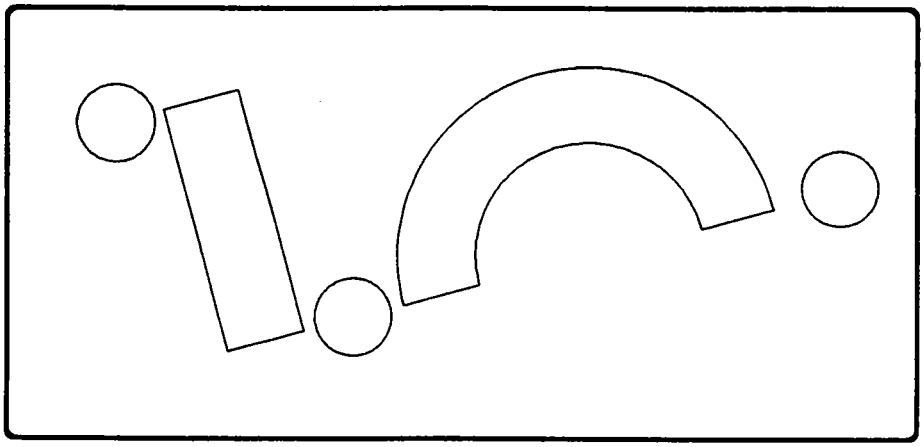


figure 2.22

Les intersections candidates témoignent toutes d'une entrée de la droite ou d'une sortie dans/de l'un de ces volumes élémentaires mais seules sont intéressantes les entrées/sorties dans la matière.

On complète alors les données en incorporant à la liste de points les intersections non encore calculées entre la droite et les volumes élémentaires (t8 sur la figure 2.23) ce qui permet, lors d'un parcours de la droite, de savoir à combien de volumes appartient la portion sur laquelle on se trouve. Pour cela, il faut savoir distinguer une entrée d'une sortie: on entre dans un volume si le produit scalaire de V (le vecteur directeur de la droite) avec N (la normale au volume au point d'intersection) est négatif, on en sort dans le cas contraire.

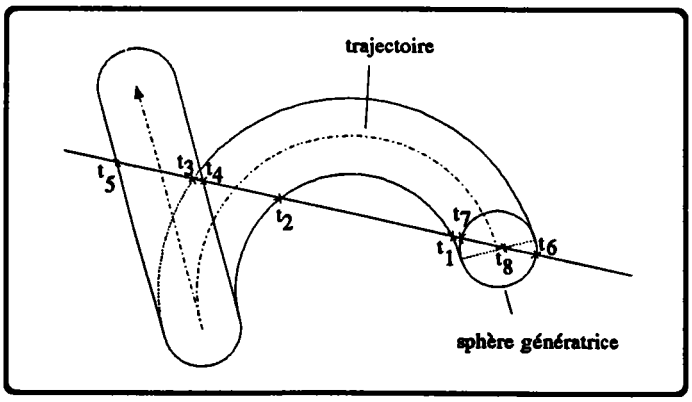


figure 2.23

On élimine alors simplement les intersections candidates inutiles: ce sont celles correspondant à des entrées alors que l'on se trouve déjà dans de la matière, ou à des sorties alors que l'on se trouve encore dans plus d'un volume élémentaire.

2.3.5. Problèmes posés dans les algorithmes annexes

Nous abordons ici le surcroît de difficulté apporté dans les techniques de restructuration par l'intégration de la fonction d'extrusion dans un modèle C.S.G.

Rappelons que ces techniques tirent parti de la propriété de non-unicité, en particulier du fait qu'une même expression booléenne peut être présentée, grâce aux lois ensemblistes (dont les lois de Morgan) de plusieurs manières équivalentes. Or de telles propriétés peuvent difficilement être mises en évidence quand interviennent des extrusions; tout au plus peut-on affirmer que:

$$\text{Extr}\ (a \cup b) = \text{Extr}\ (a) \cup \text{Extr}\ (b)$$

car

$$\text{Extr}\ (a \cup b) = \cup\ (a \cup b)_i = \cup\ (a_i \cup b_i) = (\cup a_i) \cup (\cup b_i) = \text{Extr}\ (a) \cup \text{Extr}\ (b)$$

D'une manière générale, il est faux d'écrire de semblables relations avec l'intersection et la différence (fig 2.24).

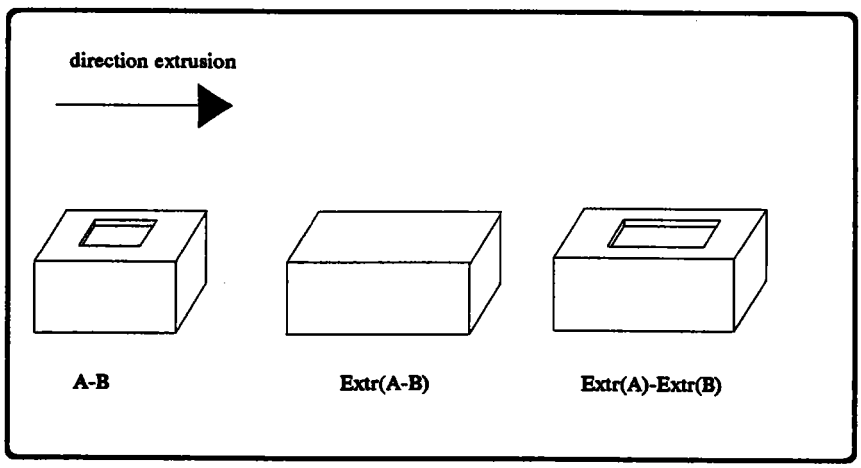


figure 2.24

La distributivité de l'extrusion par rapport à l'union est utilisée dans [PEN 83] pour calculer le volume atteignable par un robot articulé dont les bras sont approchés par des suites de sphères: le mouvement d'un bras balaye une zone qui, en vertu de la propriété ci-dessus, est égale à la réunion des zones balayées par les sphères soit, du fait des mouvements autorisés, des portions de tores et de cylindres.

2.3.6. Conversion d'une extrusion en arbre C.S.G.

Comme on l'a vu dans les paragraphes précédents, l'intégration de l'extrusion dans un arbre C.S.G. soulève de considérables problèmes: on peut donc essayer de tourner la difficulté, par exemple en exprimant une extrusion sous une forme ensembliste: l'illustration la plus simpliste en est de définir l'extrusion perpendiculaire d'un cercle comme un cylindre.

A ce sujet, Vossler [VOS 85] propose une méthode s'appliquant au déplacement d'un contour simple selon une trajectoire, orthogonale au plan du contour (prisme droit) ou circulaire (objet de révolution). Elle consiste à exprimer le contour C, que l'on suppose composé de segments et d'arcs de cercles, comme une combinaison de contours plus simples, à savoir les silhouettes de certains volumes primitifs.

Les deux opérateurs intervenant dans la décomposition sont l'union et la différence (fig 2.25).

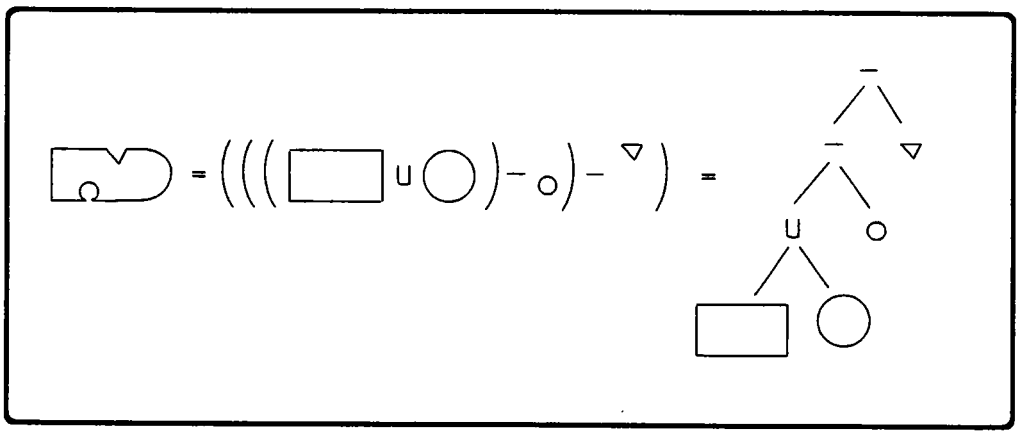


figure 2.25

Plutôt que de faire subir le mouvement à C, on va alors le faire subir à chacun des contours de la combinaison puis recomposer les volumes obtenus dans un arbre C.S.G. volumique dont la structure est une copie de l'arbre de décomposition plan: on obtient ainsi une description ensembliste du volume balayé par C (fig 2.26)

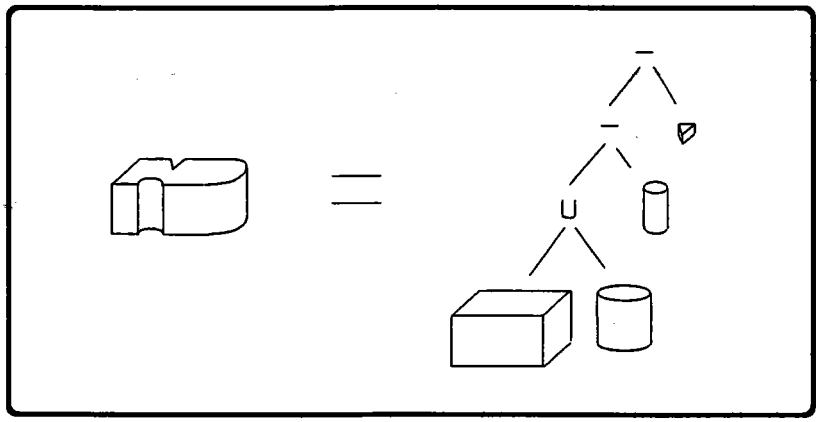


figure 2.26

La subtilité de la méthode est que les contours qui servent à la décomposition de C sont tels que leur extrusion produit des volumes primitifs (cylindre, coin, boîte ...) du fait qu'ils en sont les silhouettes; on n'utilisera donc pas les mêmes contours pour un mouvement rectiligne ou circulaire.

Telle qu'elle est présentée, cette technique n'offre une décomposition optimale ni en nombre de primitives, ni en profondeur d'arbre: celui-ci est en effet un arbre gauche, puisque l'objet est finalement exprimé sous la forme:

$$O = (...(((o_1 * o_2) * o_3) * o_4)... * o_n)$$

où O_i est une instance de primitive

et * désigne union ou différence.

Elle est également limitée à une catégorie précise de prismes et d'objets de révolution: il paraît difficile de l'étendre à d'autres prismes, d'autres objets de révolution ou d'autres contours sans multiplier le nombre de volumes primitifs. Il semble impossible d'envisager une trajectoire quelconque.

Un bref retour en arrière permet de justifier ceci: nous avons dit en 2.3.5. que l'extrusion ne peut être distribuée par rapport à la différence:

Extr (A - B) = Extr (A) - Extr (B) EST FAUX ! (1)

Or c'est pourtant bien cette propriété que Vossler applique à des contours puisque partant de l'extrusion d'un arbre, il arrive à un arbre des extrusions. Pour reprendre l'exemple précédent, on a:

Extr (rectangle + cercle1 - cercle2 - triangle)
= Extr (rectangle) + Extr (cercle1) - Extr (cercle2) - Extr (triangle)
= boite + cylindre1 - cylindre2 - coin
(en l'absence de parenthésage, on donne priorité à l'opérateur le plus à gauche)

En fait, il se trouve que, **dans ce contexte**, la relation ci-dessus est vérifiée. En effet, de par la nature du mouvement du générateur (rectiligne ou circulaire) et grâce à l'habilité de la décomposition, il n'arrive jamais qu'une instance des contours de la décomposition à un instant t n'interfère avec une autre instance, à un instant t'. Or une condition suffisante pour écrire (1) est justement que:

pour tout t < > t', A_t et B_t sont sans interférence avec $A_{t'}$ et $B_{t'}$.

Finalement, il en ressort que la méthode n'est valable que tant que l'on peut garantir que jamais une instance des contours à un instant t n'interfère avec une instance à un autre instant (en tout cas pour les contours intervenant dans une différence).

D'autres contraintes, comme la nécessité d'utiliser pour la décomposition de C des contours dont les extrusions sont des volumes primitifs, limitent encore les possibilités d'extension de la méthode.

Elle offre cependant des possibilités accrues de modélisation à un système dont le modèle est un arbre C.S.G. classique: sur un tel modèle, on choisit en principe de décrire l'objet de révolution par ses limites, à l'aide d'une approximation polyédrique par exemple. Ceci implique, s'il n'existe pas déjà, d'implanter un algorithme d'intersection rayon/ polyèdre.

Il est également intéressant de remarquer deux différences importantes avec les techniques directes de Kajiya et Van Wijk (cf 2.3.3.1 a et b) qui s'appliquent au même domaine:

- les contours qu'utilisent ces derniers appartiennent à une gamme beaucoup plus vaste, ce qui met en faveur de leur approche une puissance de description bien supérieure.
- Vossler introduit un grand nombre de primitives, donc de calculs d'intersections rayon/primitives. Qui plus est, certaines primitives se chevauchent, introduisant des volumes redondants

2.4. Modélisation par les limites

Les difficultés considérables soulevées par l'intégration d'objets extrudés dans les arbres C.S.G. ont fait que l'on s'oriente davantage, actuellement, vers un calcul explicite de la frontière du volume balayé.

Dans cette approche, on recense de nombreux travaux visant à une prise en compte mathématique, la plus exacte possible, de l'ensemble des points touchés dans le déplacement d'un générateur. Certaines ont des bases relativement anciennes, comme les surfaces réglées, d'autres, bien que leurs fondements théoriques soient établis depuis longtemps, connaissent un regain d'intérêt du fait de l'utilisation de nouveaux outils: c'est par exemple le cas avec l'intégration du calcul symbolique pour la mise en oeuvre des enveloppes.

2.4.1. Surfaces réglées

définition 1 : *Une surface réglée est telle que par chacun de ses points passe au moins une droite qui soit entièrement contenue dans la surface.*

Sous cette forme, on ne fait pas immédiatement le rapport avec les techniques d'extrusion mais le lien apparaît en donnant une définition paramétrique équivalente:

définition 2 : *Une surface réglée est définie à l'aide d'une courbe d(u) et d'une famille de vecteurs unitaires g(u):*

$$S(u,v) = d(u) + v.g(u) \qquad 0 <= u <= um \ , \ 0 <= v <= vm$$

Sous cette forme, la surface apparaît clairement comme un segment, la génératrice, orienté par $g(u)$ et se déplaçant le long de $d(u)$, la directrice.

Une dernière formulation consiste en deux directrices que l'on relie par des segments de droite, soit:

définition 3 : $S(u,v) = d1(u) + v.(d2(u) - d1(u)) \quad 0 <= u <= um \ , \ 0 <= v <= vm$

Ces surfaces, bien que d'aspect sommaire, permettent des opérations relativement variées dont quelques unes sont exposées ici.

La plus courante des ces opérations est certainement le raccord entre deux contours, par exemple lors de la création de tuyaux s'appuyant sur des sections d'aspects identiques ou variables (fig 2.27). En cela, une surface réglée est toute indiquée pour définir la face latérale d'un prisme.

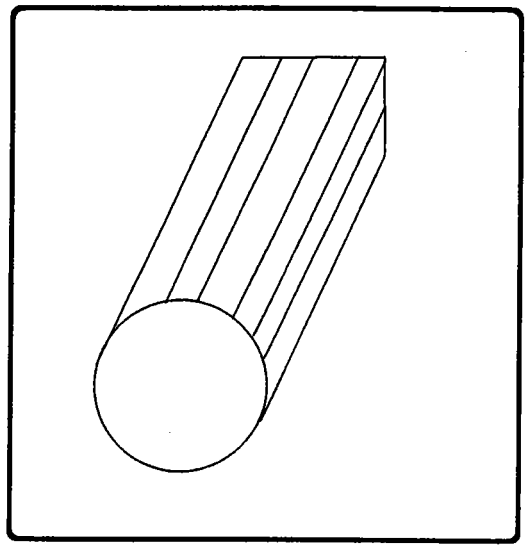


figure 2.27

D'une surface réglée, on déduit facilement une approximation polygonale s'appuyant sur les instances de la génératrice; les seules précautions à prendre sont d'imposer des segments partant des sommets quand il s'en présente (sur les directrices) et de trianguler les faces quadrangulaires non planes (fig 2.28).

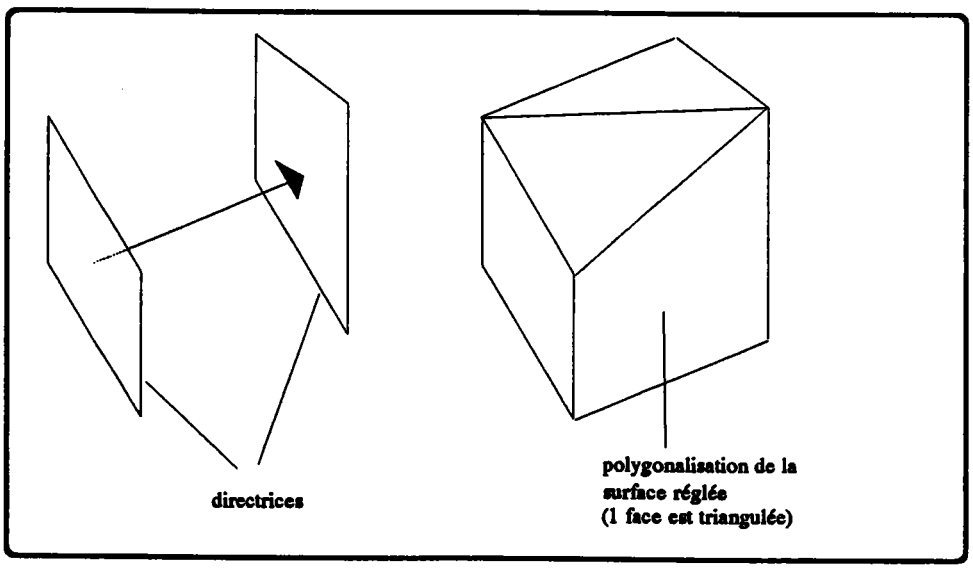


figure 2.28

Notons aussi que certaines quadriques, à savoir les cônes (fig 2.29.a), les cylindres à base elliptique, hyperbolique ou parabolique (b), les hyperboloïdes à une nappe (c) et les paraboloïdes hyperboliques (d) sont elles-mêmes des surfaces réglées. Les deux dernières sont, avec le plan, les seules à être doublement réglées [HIL 52].

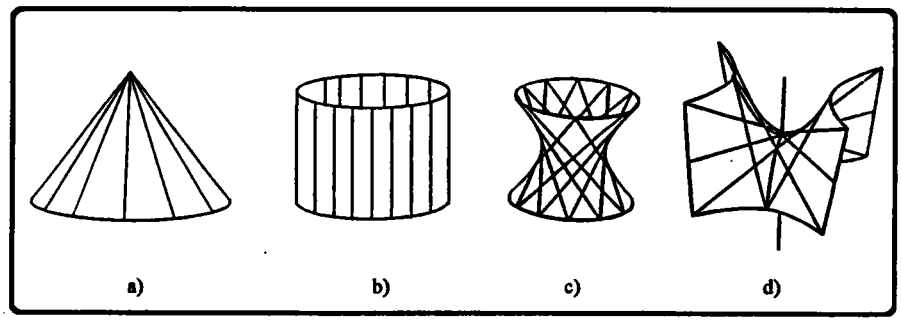


figure 2.29

Notons enfin que les surfaces réglées sont un cas particulier des surfaces polynomiales classiques (Bézier, B-splines ...). D'un degré moindre (le deuxième paramètre est de degré un), elles offrent comparativement moins de souplesse mais en contrepartie, les calculs s'en trouvent simplifiés et les imprécisions diminuées.

2.4.2. Surfaces développables

définition : *une surface développable est telle que par chacun de ses points passe au moins une droite qui soit entièrement contenue dans la surface et le long de laquelle la normale soit constante.*

Une surface développable est donc un cas particulier de surface réglée. Elle a la propriété de pouvoir être déroulée sans déformation ni déchirement sur un plan, d'où son nom. Le cône, le cylindre en font partie, la paraboloïde hyperbolique non (fig 2.30).

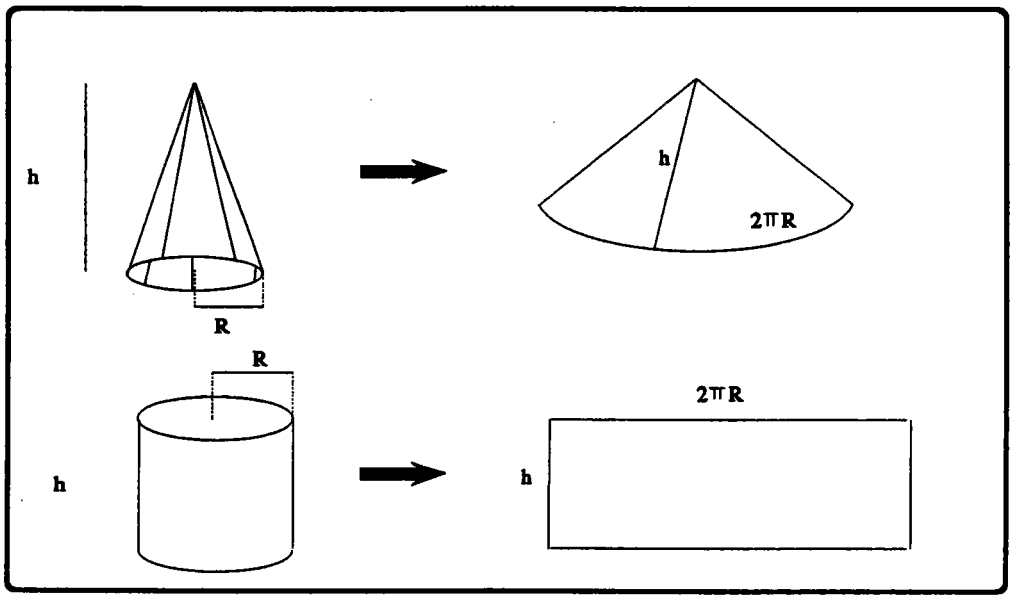


figure 2.30

Intuitivement, on comprend que ces surfaces puissent être mises en rapport avec des techniques de pliage-dépliage: en aéronautique, elles servent à calculer la "mise à plat" d'une aile d'avion et à localiser dans le plan les points de fixation correspondant à des points donnés sur la surface gauche [FAU 79].

Comme pour les surfaces réglées, on en calcule facilement une approximation bien que cette fois, cela puisse être fait plus finement avec des portions de cônes [RED 89].

On trouve dans [WEL 87] la condition pour qu'une surface réglée

$$S(u,v) = d(u) + v.g(u)$$

soit développable:

$$\frac{\partial d(u)}{\partial u} \times \left(g(u) \wedge \frac{\partial g(u)}{\partial u} \right) = 0$$

2.4.3. Théorie des enveloppes

Considérons, dans le plan, le déplacement d'un segment subissant une translation et simultanément une rotation autour de son milieu (fig 2.31).

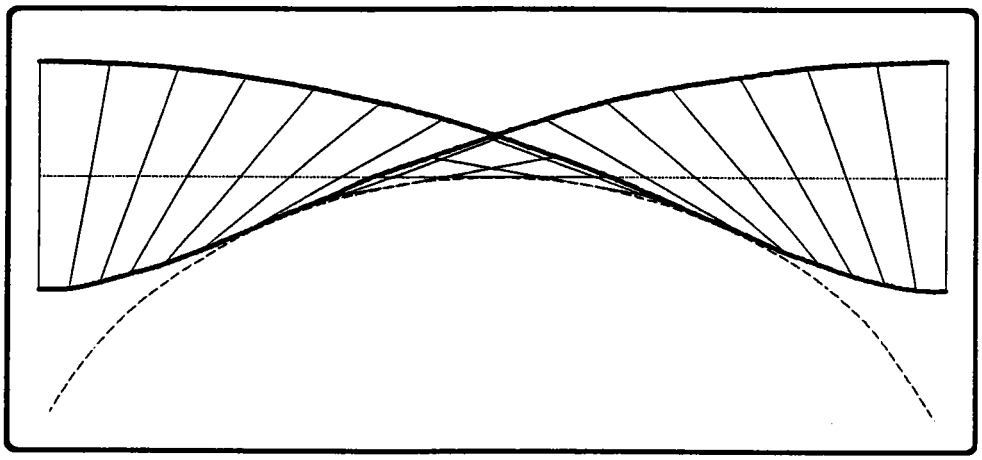


figure 2.31

On constate que la zone du plan bordée par les positions initiale et finale du segment et par les trajectoires de ses extrémités ne représente pas la totalité de la zone balayée par le segment.

Il manque une portion de courbe (en pointillés sur la figure), tangente en tout point aux instances successives du générateur et que l'on appelle l'enveloppe. Il apparaît donc qu'il faut pouvoir, dans certains cas, calculer l'enveloppe d'une famille de segments, de lignes du plan, de lignes de l'espace ou de surfaces pour limiter complètement la zone balayée par un générateur.

A cet effet, nous citons les grands traits de la théorie des enveloppes, très soigneusement étudiée et appliquée à la modélisation géométrique dans [WEL 87], puis exposons quelques uns des usages qui en sont faits.

2.4.3.1. Formulation implicite de l'enveloppe

On représente une famille de courbes du plan par l'équation implicite

$$f(x,y,t) = 0. \qquad (1)$$

Pour un t donné, généralement dans l'intervalle [0,1], l'équation (1) devient celle de la $t^{\text{ième}}$ courbe de la famille.

Cette équation caractérise aussi les positions successives, fonction du temps, d'une courbe subissant un déplacement.

Si elle existe, l'enveloppe d'une telle famille de courbes se calcule traditionnellement de la manière suivante:

- considérons un instant $t=t_0$ donné.
- on remarque que les courbes correspondant aux instants t_0 et t_0+eps ont des intersections qui se rapprochent d'autant plus de l'enveloppe que eps est petit (fig 2.32).

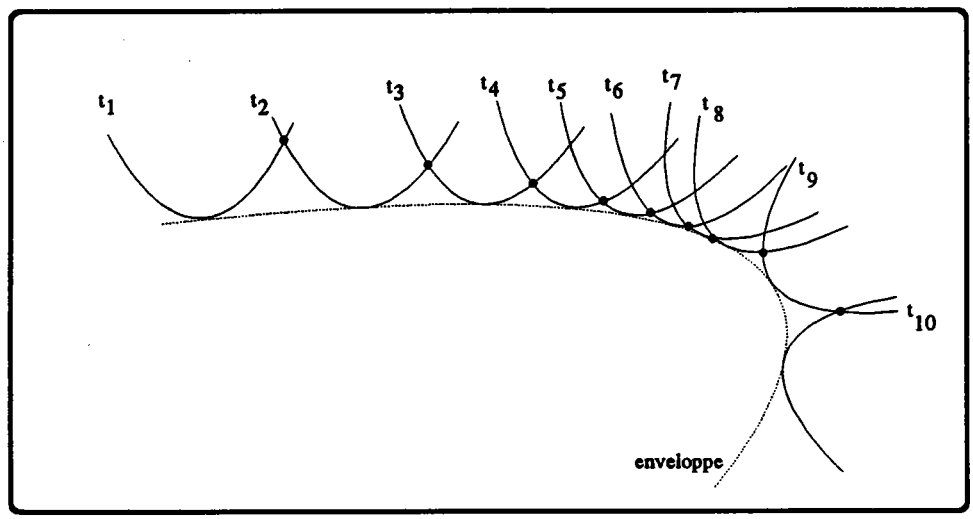


figure 2.32

- on en tire que les intersections des deux courbes sont sur l'enveloppe quand eps tend vers 0 et, par conséquent, que les points de l'enveloppe vérifient:

$$f(x,y,t_0) = 0$$

et $f(x,y,t_0+eps) = 0$ pour eps tendant vers 0

- en combinant ces deux dernières équations, on trouve

$$f(x,y,t_0) = 0$$

et $$\frac{f(x,y,t_0+eps) - f(x,y,t_0)}{eps} = 0 \qquad \text{quand eps} \dashrightarrow 0$$

- finalement, aucune hypothèse n'étant faite sur t_0, on caractérise l'enveloppe d'une famille $f(x,y,t) = 0$ de courbes du plan par:

$$f(x,y,t)=0$$

et

$$\frac{\partial f}{\partial t}(x,y,t)=0$$

D'une manière tout-à-fait comparable, l'enveloppe d'une famille f (x,y,z,t) de surfaces de $\mathbf{R}^3$ est caractérisée par:

$$f(x,y,z,t)=0$$

et

$$\frac{\partial f}{\partial t}(x,y,z,t)=0$$

Pour mettre en évidence l'équation implicite de l'enveloppe, on se sert généralement de la deuxième équation pour tirer une expression de t que l'on substitue dans la première équation.

On obtient ainsi l'équation (sans t) du discriminant de la famille, discriminant qui *contient* l'enveloppe cherchée: si f porte des points singuliers, des points doubles, des points isolés ou des courbes singulières (dans le cas d'une surface), si une portion de f est au contact d'une autre portion, alors le discriminant comprend aussi la famille de ces points ou lignes singuliers.

Exemple 1 :

Soit une famille de cercles du plan centrés sur la première bissectrice et tangents aux deux axes (fig 2.33):

$$(x-t)^2 + (y-t)^2 = t^2 \qquad (5)$$

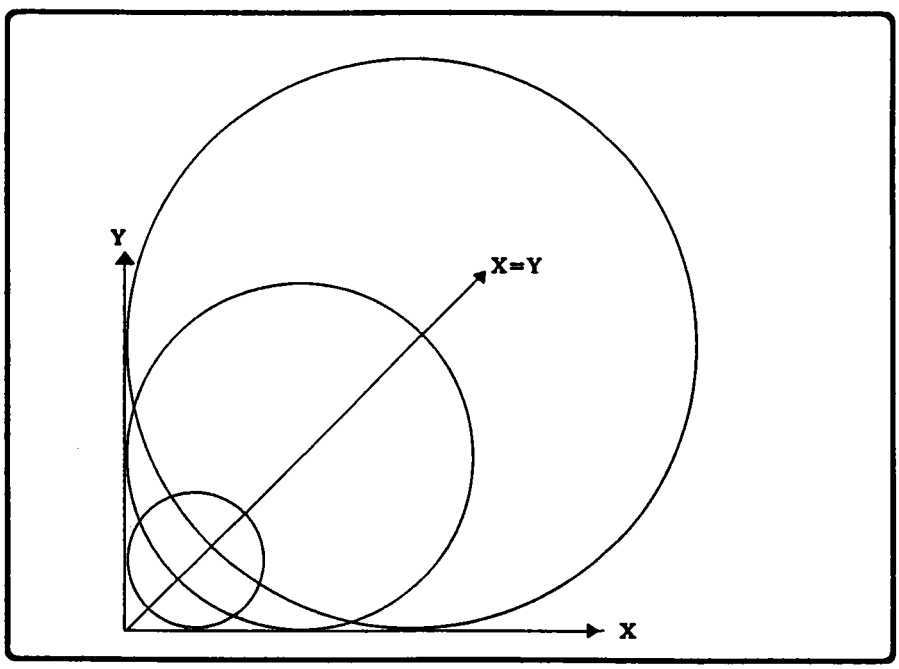


figure 2.33

La dérivée par rapport à t de cette expression s'annule si $t = x+y$.
L'expression (5) devient alors

$$2.x.y = 0 \quad \text{soit} \quad x = 0 \text{ ou } y = 0.$$

L'enveloppe de la famille des cercles, ici égale au discriminant, est donc constituée de deux droites, (Ox) et (Oy).

Exemple 2 :

Soit une famille de courbes du plan (fig 2.34)

$$y = (x-t)^3$$

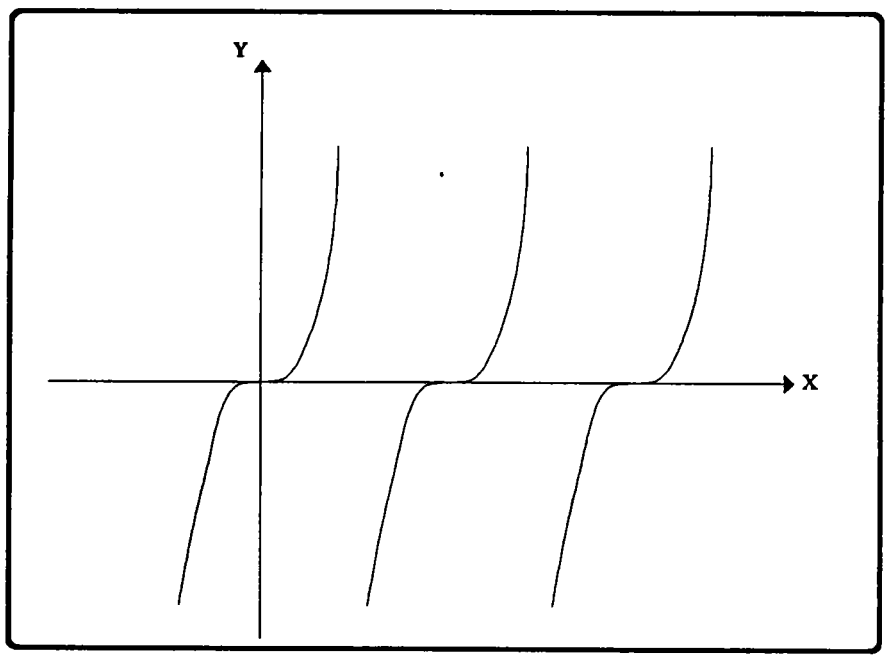


figure 2.34

La dérivée par rapport au temps est $-3.(x-t)^2$, qui s'annule pour $t = x$. Par substitution dans l'équation ci-dessus, on trouve l'équation de l'enveloppe:

$$y = 0.$$

On constate que l'existence d'une enveloppe n'implique pas que la famille de courbes admette une limite géographique. Nous reviendrons sur ce point au moment de fermer la section consacrée aux enveloppes.

Exemple 3 :

Soit une sphère de rayon variable se déplaçant le long de l'axe (0x) (fig 2.35):

$$(x-t)^2 + y^2 + z^2 = t^2/2.$$

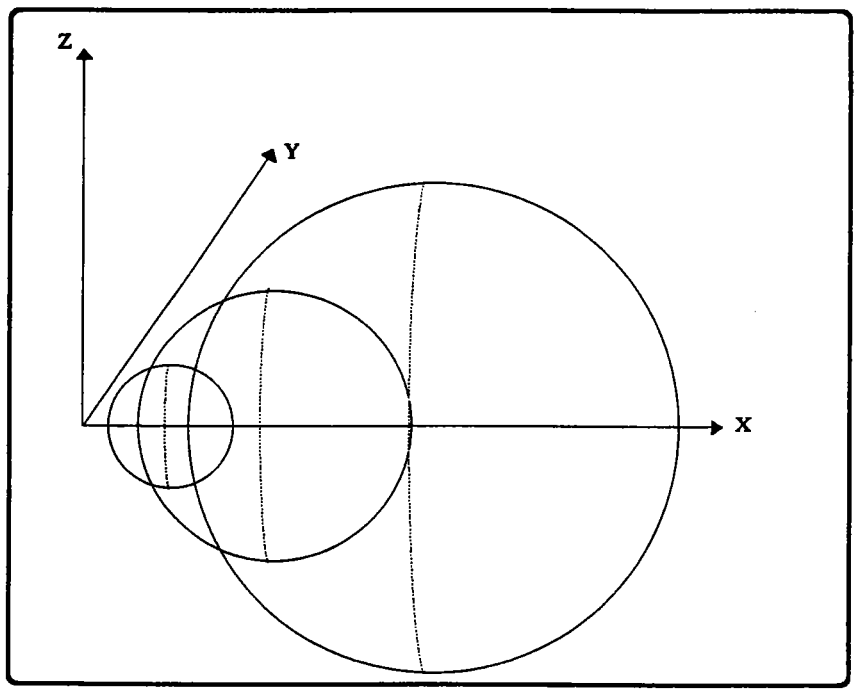


figure 2.35

La dérivée par rapport au temps ($t - 2.x = 0$) s'annule pour $t = 2.x$. Par substitution dans l'équation ci-dessus, on trouve l'équation de l'enveloppe:

$$y^2 + z^2 = x^2$$

soit un cône d'axe (Ox), de sommet O et d'angle au sommet 45 degrés.

Ces trois exemples ont été choisis tels que les équations puissent être combinées de manière à supprimer t. Il suffit d'essayer quelques cas au hasard pour se convaincre de la difficulté du problème dans le cas général.

2.4.3.2. Formulation paramétrique de l'enveloppe

Lorsque la famille des instances du générateur est exprimée paramétriquement, la condition nécessaire pour l'existence d'une enveloppe porte sur le jacobien de l'application associant à un n-uplet de paramètres les équations paramétriques du mouvement [WEL 87].

Pour ne retenir que les résultats essentiels, disons que:

- l'enveloppe d'une famille de courbes

$$f(u,t) = [x(u,t),y(u,t)] \qquad (2)$$

de $\mathbf{R}^2$ existe si

$$\begin{vmatrix} \frac{\partial x}{\partial u} & \frac{\partial x}{\partial t} \\ \frac{\partial y}{\partial u} & \frac{\partial x}{\partial t} \end{vmatrix} = 0$$

On la calcule en exprimant u en fonction de t à l'aide de (1) puis en substituant dans (2); les équations que l'on en tire sont de la forme $[e_1(t),e_2(t)]$: l'enveloppe est donc elle-même une courbe de $\mathbf{R}^2$.

- de même, la condition nécessaire à l'existence d'une enveloppe pour la famille $[x(u,t),y(u,t),z(u,t)]$ de courbes de l'espace est:

$$\left(\frac{\partial x}{\partial u} \; \frac{\partial y}{\partial u} \; \frac{\partial z}{\partial u}\right) \wedge \left(\frac{\partial x}{\partial t} \; \frac{\partial y}{\partial t} \; \frac{\partial z}{\partial t}\right) = 0$$

Ceci permet d'exprimer u en fonction de t puis de substituer dans les équations paramétriques de la famille; l'enveloppe ainsi mise en évidence est sous la forme $[e_1(t),e_2(t),e_3(t)]$: elle est donc elle-même une courbe de $\mathbf{R}^3$.

- la condition nécessaire à l'existence d'une enveloppe pour la famille $[x(u,v,t),y(u,v,t),z(u,v,t)]$ de surfaces est

$$\begin{vmatrix} \frac{\partial x}{\partial u} & \frac{\partial x}{\partial v} & \frac{\partial x}{\partial t} \\ \frac{\partial y}{\partial u} & \frac{\partial y}{\partial v} & \frac{\partial y}{\partial t} \\ \frac{\partial z}{\partial u} & \frac{\partial z}{\partial v} & \frac{\partial z}{\partial t} \end{vmatrix} = 0$$

Ceci permet d'exprimer u ou v en fonction de t et donc, par substitution dans les équations paramétriques de la famille, d'obtenir les équations paramétriques de l'enveloppe sous la forme

$$[e_1(u,t),e_2(u,t),e_3(u,t)] \quad \text{ou} \quad [e_1(v,t),e_2(v,t),e_3(v,t)]$$

suivant que l'on a exprimé v ou u en fonction de t. L'enveloppe est donc elle-même une surface.

Exemple 1 :

Soit un segment du plan de longueur 2,initialement parallèle à (O,y) et à une distance d de O. Il subit une rotation autour de O,sa médiatrice passe en permanence par O et il s'éloigne de O au fur et à mesure qu'il tourne (on peut imaginer qu'il est attaché à une ficelle enroulée autour de (Oz): au fur et à mesure qu'il tourne, la ficelle se déroule et il s'éloigne) (fig 2.36.a).

La famille de segments est définie par:

$$x(u,t) = d.(1+t).\cos(2.pi.t) - u.\sin(2.pi.t) \qquad 0 <= t <= 1$$
$$y(u,t) = d.(1+t).\sin(2.pi.t) - u.\cos(2.pi.t) \qquad -1 <= u <= 1$$

Après simplifications, l'expression sur les dérivées partielles donne

$$d - 2.pi.u = 0 \text{ soit } u = d/(2.pi)$$

Par substitution, on tire donc les équations paramétriques de l'enveloppe:

$$x(t) = d.(1+t).\cos(2.pi.t) - d.\sin(2.pi.t)/(2.pi) \qquad 0 <= t <= 1$$
$$y(t) = d.(1+t).\sin(2.pi.t) - d.\cos(2.pi.t)/(2.pi) \qquad -1 <= u <= 1$$

soit une spirale centrée sur O et dont le rayon, à chaque tour, augmente de d (fig 2.36.b).

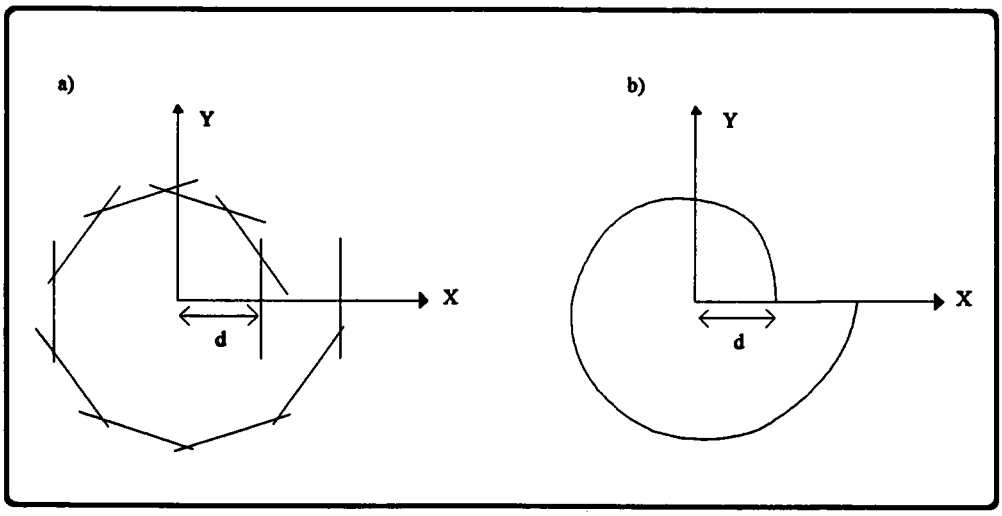


figure 2.36

Exemple 2 :

Soit un plan parallèle à (Oz), distant de d et subissant une rotation autour de (Oz) (fig 2.37). La famille de plan est:

$$x(u,v,a) = u.\sin(a) + d.\cos(a)$$
$$y(u,v,a) = -u.\cos(a) + d.\sin(a)$$
$$z(u,v,a) = v$$

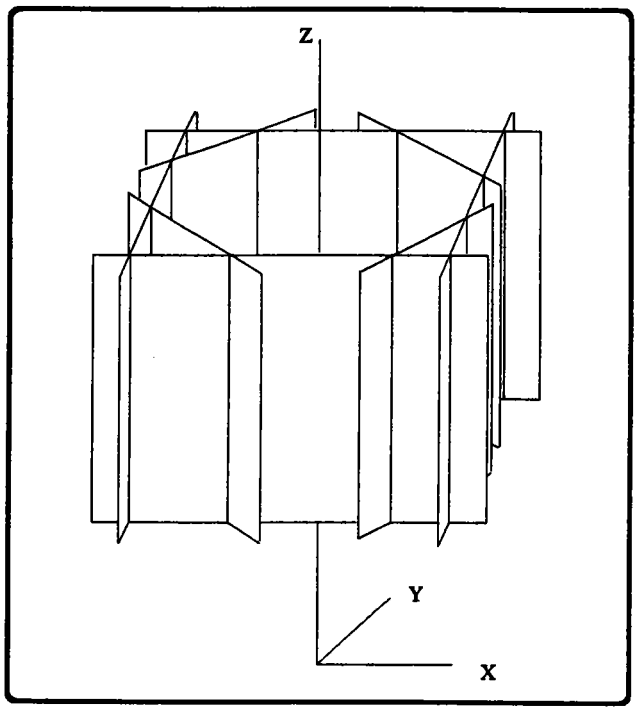


figure 2.37

Le déterminant 3x3 évoqué plus haut s'annule pour u = 0. Les équations de l'enveloppe sont alors:

$$x(v,a) = d.\cos(a)$$
$$y(v,a) = d.\sin(a)$$
$$z = v$$

Il s'agit donc d'un cylindre d'axe (Oz) et de rayon d. Lorsque d=0, l'enveloppe, supposée être une surface, dégénère en une droite (axe 0z).

L'application de ces mêmes principes à une famille de plans Pl(u,v,t) = A(t) + u.e1(t) + v.e2(t), passant par un point A(t) et orientés par [e1(t),e2(t)], montre que, sauf exception, l'enveloppe de la famille est une surface développable d'équation

$$S(t,v)=A-e_1 \frac{A'x(e_1 \wedge e_2)}{e'_1\, x(e_1 \wedge e_2)} + v \left(e_2 - e_1 \cdot \frac{e'_2\, x(e_1 \wedge e_2)}{e'_1\, x(e_1 \wedge e_2)} \right)$$

Cette notion correspond bien à l'intuition que l'on peut avoir de l'enveloppe d'une famille de plans: en effet, par définition, l'enveloppe s'appuie sur l'intersection entre deux éléments consécutifs de la famille, ici une droite. En considérant des paires successives de plans, on engendre une série de droites qui constituent les instances de la génératrice de la surface développable. Chaque plan de la famille est tangent à la surface sur toute la longueur d'une génératrice.

Les exceptions évoquées dans la définition sont au nombre de quatre:

- si, pour tout t, e'1(t) = e'2(t)=0 (l'apostrophe désigne la différentiation par rapport au temps) alors tous les plans sont parallèles. Il n'ont une enveloppe que s'ils sont tous confondus (A' x (r1 ^ r2) = 0) et celle-ci est alors le plan lui-même.
- si seul e'1(t) ou e'2(t) est identiquement nul, l'enveloppe existe et est un cylindre dont l'axe est orienté par celui des deux vecteurs qui est constant.
- l'un des vecteurs e'1(t) ou e'2(t) est tout le temps dans le plan correspondant à l'instant t: le deuxième vecteur l'est alors forcément aussi: tous les plans de la famille sont confondus et constituent l'enveloppe.
- A(t) est constant si bien que les plans ont tous un point commun: il n'y a d'enveloppe que s'ils sont tous superposés (e'1 x (e1 ^ e2) = e'2 x (e1 ^ e2) = 0) et celle-ci est alors égale au plan lui-même.

2.4.3.3. Autre caractérisation de l'enveloppe

Une autre caractérisation de l'enveloppe est qu'un point P du générateur Gt n'appartient aussi à l'enveloppe que si son vecteur vitesse est sur la droite -ou le plan- tangent à Gt en P, autrement dit, si la vitesse est perpendiculaire à la normale à Gt en P:

$$V(p).N(p) = 0$$

Plusieurs applications utilisant ce principe sont citées dans la littérature: c'est le cas de [GHO 84], où l'on calcule la zone de l'écran balayée par une brosse (ou une gomme); il s'agit là d'une utilisation D.A.O. Sur le même principe, l'exemple cité en 2.4.4.4. concerne la simulation de l'usinage [WAN 86].

2.4.4. Application de la théorie des enveloppes à l'extrusion

2.4.4.1. Propriétés utilisées par toutes les méthodes

Nous citons dans les paragraphes qui suivent quelques unes des utilisations des enveloppes. Dans la plupart des cas, on utilise des règles qui amènent à raisonner sur des éléments de moindre dimension. Nous en disons quelques mots dans les lignes qui suivent.

Soit une zone Z du plan, bordée par un contour C. Le déplacement de Z dans le plan engendre une surface S = Extr (Z) et on montre [WEL 87] que:

$$S = Extr(Z) = Z \cup Extr(C)$$

Autrement dit, l'ensemble des points balayés par Z est le même que celui balayé par sa frontière sauf si l'amplitude du mouvement est trop "petite": c'est la raison pour laquelle Z intervient dans la formule ci-dessus (fig 2.38).

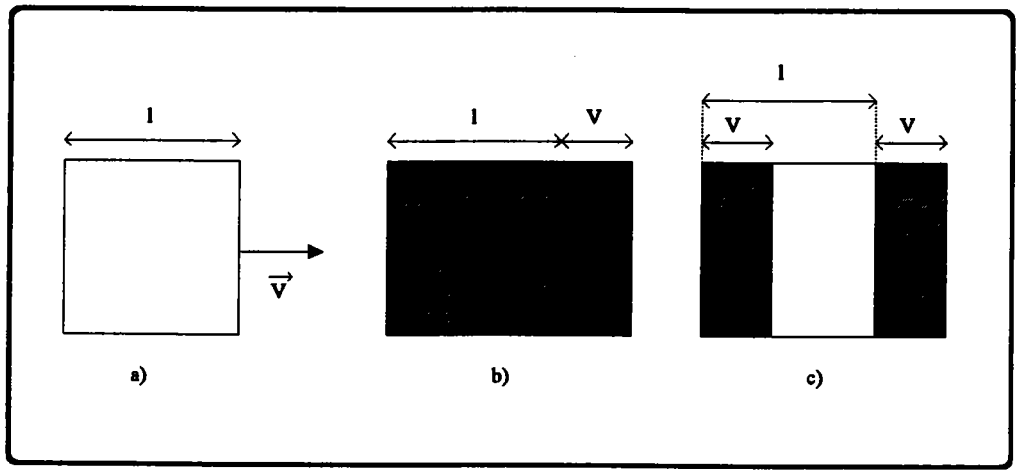


figure 2.38: importance de l'amplitude du mouvement:

a) le générateur (surface rectangulaire) et le vecteur d'extrusion

b) zone balayée par le générateur

c) zone balayée par la frontière du générateur

Dans le cas contraire, on peut se passer de Z, mais il est difficile, sauf dans le cas de mouvements simples, d'établir que l'amplitude du mouvement est trop petite; par contre, on exhibe facilement une condition *suffisante* pour affirmer que le mouvement est assez ample pour que Z soit négligé:

$$\text{si } Z_0 \cap Z_1 = \varnothing \quad \text{alors } S = \text{Extr}(Z) = \text{Extr}(C)$$

Une condition *nécessaire et suffisante*, cette fois, est qu'il existe une instance Zt du générateur qui soit disjointe de Z. Le mouvement étant continu, on est alors sûr que les points de Z ont été balayés au moins une fois dans l'intervalle de temps [0,t]; plus formellement:

$$\text{si } \exists\, t \in [0,1] \;/\; Z_t \cap Z = \varnothing \text{ alors } S = \text{Extr}(Z) = \text{Extr}(C) \qquad (1)$$

On ramène donc l'extrusion d'une surface plane à l'extrusion de son bord. Le bord est lui-même constitué d'un certain nombre d'arêtes, droites ou courbes, donc peut être défini comme la réunion de ces arêtes:

$$C = \bigcup_{1}^{nbseg} A_i$$

En distribuant, on en déduit que

$$\text{Extr}(C) = \text{Extr}\left(\bigcup_i A_i \right) = \bigcup_i \text{Extr}(A_i)$$

c'est-à-dire que la zone balayée par C est l'union des zones balayées par ses arêtes. Finalement, on a:

$$\begin{aligned} &Extr(Z)=Z \cup [\bigcup_i Extr(A_i)] && \text{si} \nexists\, t \;/\; Z_t \cap Z = \varnothing \\ &Extr(Z)= \bigcup_i Extr(A_i) && \text{si} \exists\, t \;/\; Z_t \cap Z = \varnothing \end{aligned} \qquad (2)$$

On réduit donc la difficulté en exprimant l'extrusion d'une zone du plan à l'aide des extrusions des arêtes de son bord.

En modélisation cependant, on manipule plus facilement les limites d'un ensemble de points que l'ensemble lui-même. Terminons donc avec la détermination du bord de la zone balayée, Fr (Extr (Z)).

Il est clair, d'après (2), que cette ligne est composée de segments de la frontière de Z et de segments des frontières des zones balayées par les arêtes (la frontière de A U B ne comprend que des portions de celles de A et de B):

$$Fr(Extr(Z)) \subset Fr(Z) \cup [\bigcup_i Fr(Extr(A_i))]$$

Considérons une arête A_i de Fr (Z) = C. Dans la plupart des cas, A_i participe à $Fr(Extr(A_i))$; les exceptions viennent de ce que Extr (A_i) peut se recouper et, du fait, A_i peut, partiellement ou en totalité, y être incluse. Si elle l'est en totalité, elle ne présente plus d'intérêt en tant que frontière; sinon la portion extérieure participe à Fr (Extr (A_i)).

On en conclut donc que toute arête A_i de Fr (Z) est soit:

- totalement incluse dans Extr (A_i) et elle ne peut participer à Fr (Extr (Z)),
- non totalement incluse dans Extr (A_i) et sa partie utile (extérieure) appartient à Fr(Extr (A_i)),

c'est-à-dire que

$$Fr(Z) \subset \bigcup_i Fr(Extr(A_i))$$

d'où

$$Fr(Extr(Z)) \subset \bigcup_i Fr(Extr(A_i))$$

Tous ces résultats peuvent être étendus à l'extrusion d'un volume V bordé par les faces F_i: on a

$$\begin{aligned} Extr(V) &= V \cup Extr(Fr(V)) \\ &= Extr(Fr(V)) \quad si \nexists\, t \,/\, V_t \cap V = \varnothing \\ &= \bigcup_i Extr(F_i) \quad si \nexists\, t \,/\, V_t \cap V = \varnothing \end{aligned}$$

$$et\ Fr(Extr(V)) \subset \bigcup_i Fr(Extr(F_i))$$

2.4.4.2. Aire balayée par un polygone dans le plan

Nous citons ici les travaux de Sambandan, Kedem et Wang [SAM 89] sur le calcul des bords de la zone du plan balayée par un polygone en mouvement ne subissant aucune déformation.

Un moyen de faire serait de procéder comme le suggère l'exemple du segment donné en 2.4.3.: la zone balayée par ce dernier est limitée par des portions de l'enveloppe, des portions des trajectoires des deux extrémités et par ses positions initiale et finale. Par conséquent, la frontière de la zone ZB touchée par le polygone est constituée de parties des courbes engendrées avec ce principe par chacun des segments (d'après la règle sur la frontière de l'extrusion d'une zone du plan).

Les auteurs visent toutefois à diminuer le nombre de courbes afin de faciliter les choix. Leur méthode consiste dans un premier temps à déterminer, sur un certain nombre d'instances Gt du générateur, les points susceptibles d'appartenir à Fr (ZB).

Soit un instant t donné. Comme le mouvement est rigide, il existe un point Ct, appelé centre de rotation instantané, en regard duquel le mouvement peut être réduit à une rotation pure. On définit une application "*distance au centre*" qui, à tout point de Gt, associe sa distance à Ct:

$$\begin{aligned} d\ : G_t &\longrightarrow R \\ P &\longrightarrow d(P,C_t) \end{aligned}$$

On sait alors que les points auxquels l'application d connaît un extremum sont susceptibles d'appartenir à Fr (ZB). Ils n'en sont que susceptibles parce que tout point de Fr (ZB) constitue effectivement un extremum mais tout extremum ne participe pas forcément à la frontière (fig 2.39).

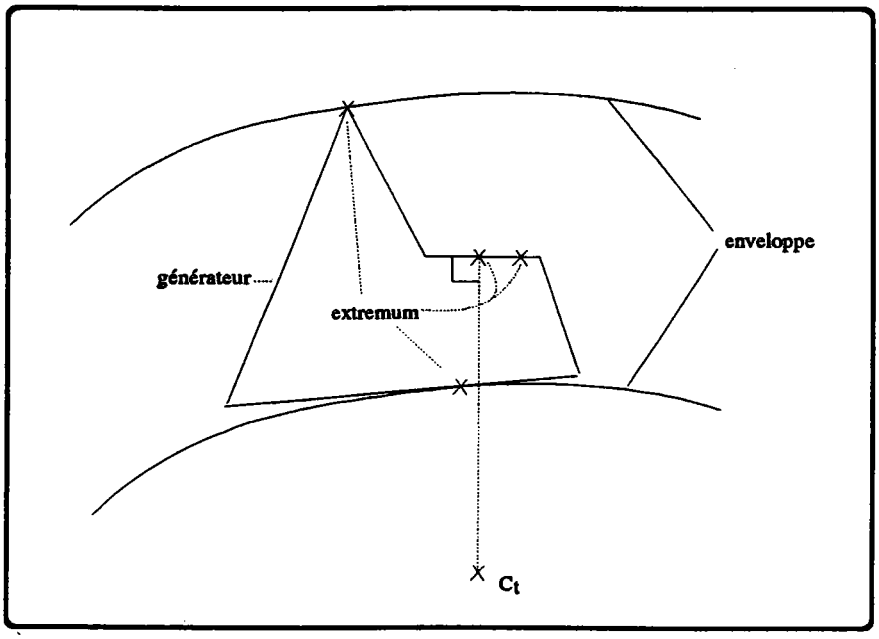


figure 2.39

(Cette notion rejoint celle des contours extrêmes qu'utilise Korein [KOR 85] pour le calcul du volume balayé par un polyèdre)

Un extremum appartient à deux catégories: il s'agit soit d'un sommet, que l'on dit *actif*, soit d'un point sur une arête: dans ce dernier cas, le point est simplement le projeté de Ct sur l'arête et cette dernière est dite *active*.

Dans un deuxième temps, on recense ceux des sommets et celles des arêtes qui sont respectivement actifs et actives pendant plusieurs instants consécutifs; les autres ne nous intéressent plus.

Les sommets actifs engendrent des courbes susceptibles de participer à Fr (ZB), de même que les points caractéristiques des arêtes actives. Il faut remarquer qu'un point caractéristique n'est pas fixe mais qu'au contraire il glisse sur l'arête: quand celle-ci cesse d'être active, c'est forcément que le point est arrivé à une extrémité ce qui permet de calculer précisemment l'instant t correspondant: c'est celui auquel Ct se projette sur l'extrémité de l'arête. Grâce à ceci, on limite exactement la portion utile de la courbe engendrée par le projeté de Ct sur l'arête active (la courbe en question est tout simplement l'enveloppe des positions successives de l'arête, bien que cela ne soit pas mentionné expressément).

Finalement, on en déduit un certain nombre de courbes que l'on complète avec les polygones de départ et d'arrivée.

La frontière de ZB est alors reconstituée en parcourant les courbes dans un certain sens (en laissant la matière à gauche) et, à une intersection, en prenant toujours la première à droite.

2.4.4.3. Extrusion de polyèdres

Le problème du calcul de la zone de l'espace balayée par un volume n'est pas traité dans sa généralité. Couramment, on approche le générateur par un polyèdre légèrement surdimensionné et c'est lui qui subit le mouvement.

Ce paragraphe donne donc les grandes lignes de la méthode proposée par Weld [WEL 87] pour l'extrusion de polyèdres. Sur ce thème, on peut aussi consulter [KOR 85] où l'on applique l'extrusion de polyèdres au calcul du volume atteignable d'un robot.

Nous appellerons G le polyèdre générateur, ZB la zone qu'il balaie et Fr (ZB) la frontière de cette zone.

Fr (ZB) est constituée de portions des frontières des zones balayées par les faces de G. Le problème du calcul de l'extrusion d'un polygone se pose donc à nouveau, avec cette fois une trajectoire quelconque, si bien que l'objet engendré est un volume (fig 2.40.d).

Celui-ci est bordé par deux ou trois types de surfaces, suivant la trajectoire:
- d'abord les première et dernière instances du générateur (soit deux polygones).
- ensuite les surfaces réglées engendrées par les arêtes de la face (fig 2.40 b et c).
- enfin, sous certaines conditions, un segment de l'enveloppe de la famille des plans portant les instances successives de la face (fig 2.40.a).

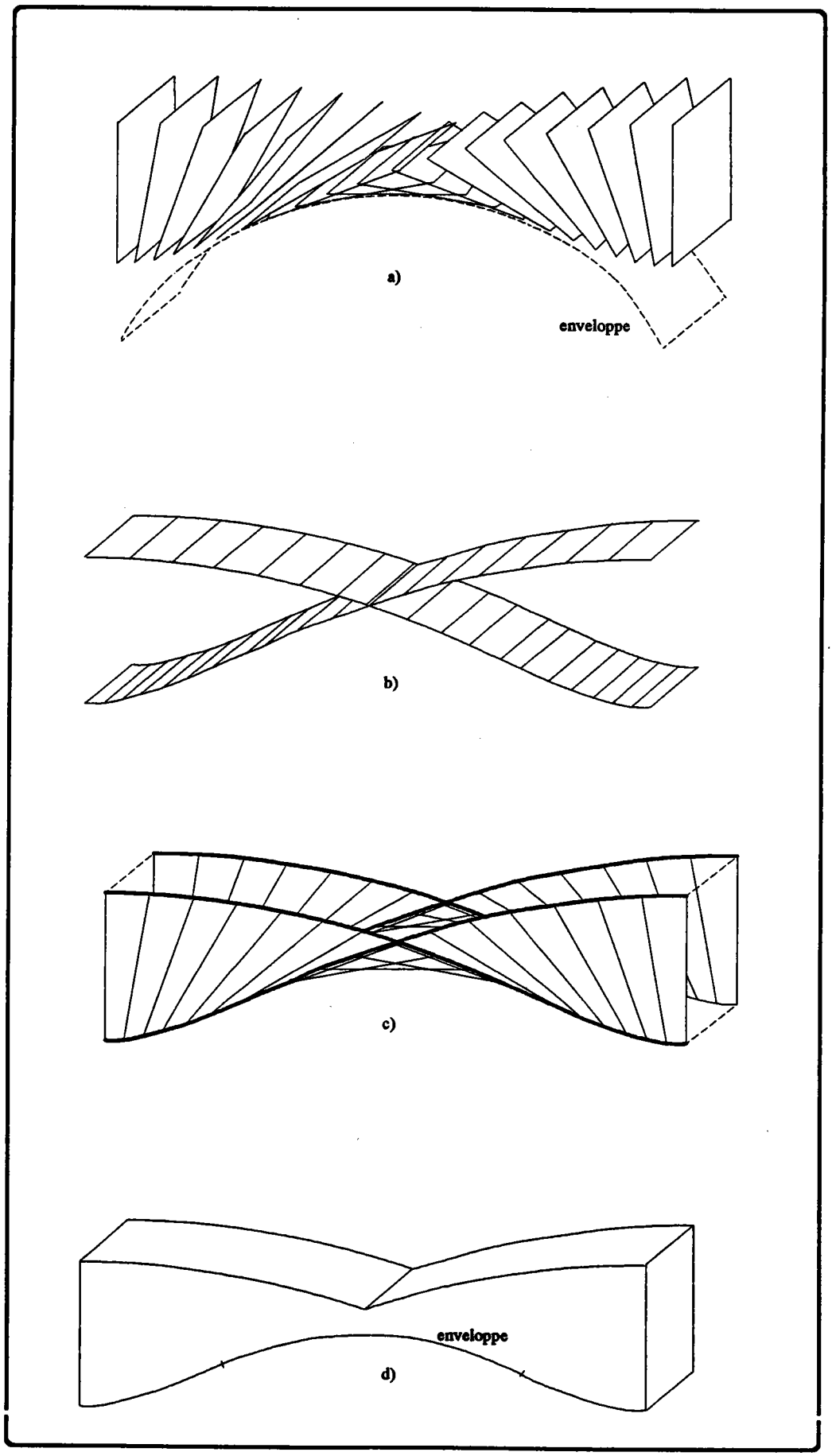


figure 2.40

On constate que les surfaces réglées engendrées par deux arêtes consécutives de la face sont connectées le long de la trajectoire de leur sommet commun; il en va de même pour les surfaces engendrées par les autres arêtes: dans le meilleur des cas, on construit donc immédiatement les limites du volume balayé par la face en prenant ses positions initiale et finale et le "tuyau" engendré par la suite de ses arêtes.

Mais si le mouvement est tel que des instances successives de la face se recoupent, alors:

- les surfaces réglées se recoupent aussi, si bien que certaines portions, maintenant inutiles, se trouvent à l'intérieur du volume et sont à retirer. Les points de ces surfaces se classent donc en deux catégories:
  - . ceux participant à la frontière du volume engendré, auxquels on peut associer une normale extérieure.
  - . ceux se trouvant à l'intérieur de ce même volume et auxquels on ne peut associer de normale.
- il faut introduire une frontière supplémentaire, à savoir une portion de l'enveloppe des plans portant les instances de la face. En vertu de ce qui a été dit en 2.4.3.2., l'enveloppe est une surface développable. La limite de la portion utile est une courbe fermée, résultat de l'intersection du tuyau des surfaces réglées avec l'enveloppe; elle est calculée de manière approchée.

Les volumes élémentaires engendrés par les faces du polyèdre étant construits, la frontière du volume balayé par le polyèdre peut être évaluée: on remarque pour cela que chaque arête appartient à deux faces exactement. La surface réglée qu'elle engendre participe donc, en totalité ou partiellement, à la frontière de deux volumes. Pour savoir quelles portions de cette surface interviennent dans Fr (ZB), on applique à chacun de ses points la règle suivante:

soient n1 et n2 les deux normales associées au point P, l'une relative au premier volume, l'autre au deuxième.

si n1 ou n2 n'est pas définie **alors**
P n'appartient pas Fr (ZB)
sinon { n1 = n2 ou n1 = -n2 }
p est sur Fr (ZB) **ssi** (n1 = n2) autrement dit
ssi (il n'y a de la matière que d'un coté de P)
fsi

Les surfaces développables intervenant dans les frontières des volumes engendrés par les faces n'ont pas la propriété de pouvoir être appariées. Elles doivent donc toutes être traitées séparément mais Weld ne donne pas d'explications sur la manière de faire. Une solution peut être de ne conserver, pour la surface développable engendrée par une face donnée, que les portions qui n'appartiennent à aucun des volumes engendrés par les autres faces.

2.4.4.4. Application à l'usinage

La dernière application des enveloppes citée ici concerne la simulation de l'usinage [WAN 86]. Un cylindre de rayon r, symbolisant une fraise, se déplace le long d'une courbe. Son mouvement est défini:

- par la trajectoire C(t) du centre de la base
- par un vecteur unitaire A(t) donnant l'orientation de son axe.

Dans la suite, nous omettrons de préciser la dépendance des fonctions vis-à-vis de t: on notera donc A pour A(t), C pour C(t)... On convient également que l'apostrophe (') désigne la différenciation par rapport au temps.

A partir de A(t), on définit un repère local par:

- e1 (t) = A
- e2 (t) = A' / ||A'||
- e3 (t) = A ^ A' / ||A'||

Pour cela, il faut supposer non constante l'orientation du cylindre (||A'|| <> 0). Un deuxième repère local est proposé pour ce cas précis mais il n'ajoute rien à l'intérêt de la méthode; nous n'en parlons donc pas ici.

Les coordonnées d'un point du cylindre sont fonction de trois paramètres (fig 2.41):

$$P(t,v,a) = C + v.A + r.\cos(a).e2 + r.\sin(a).e3 \qquad (1)$$

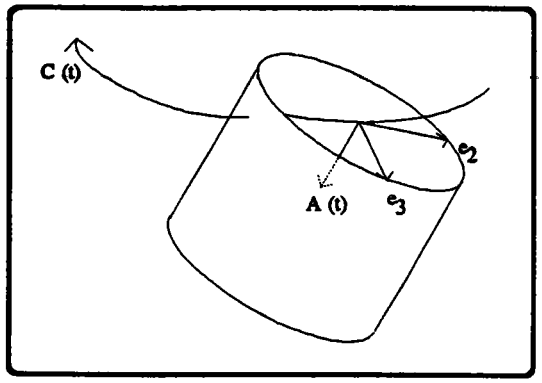


figure 2.41

Sa vitesse est obtenue en dérivant par rapport au temps et en faisant intervenir d'assez complexes équations de cinématique:

$$V(P) = C' + v.||A'||.e2 - r.\cos(a).||A'||.e1$$

Sa normale est donnée simplement par

$$N(P) = \cos(a).e2 + \sin(a).e3$$

Nous savons que pour que P appartienne à l'enveloppe de la famille de cylindres, sa vitesse doit être perpendiculaire à la normale, soit

$$N(p) \times V(P) = 0$$

ou $\cos(a).(e2 \times C') = \cos(a).v.||A'|| + \sin(a).(e3 \times C') = 0$

ou encore $a = tg^{-1}\left[\dfrac{e2 \times C' + v.|\,|A'|\,|}{e3 \times C'} \right]$

On exprime ainsi a en fonction de t et v et par substitution dans (1), on tire les équations bi-paramétriques de l'enveloppe.

Il est clair cependant que la formule obtenue n'est guère pratique et qu'en particulier les intersections avec un rayon risquent d'être soumises à des imprécisions non négligeables.

En conséquence, Wang et Wang approchent le volume engendré par un polyèdre et c'est ce dernier que l'on retranche au brut pour simuler l'usinage. Le résultat de la soustraction n'est pas évalué mais directement affiché par ray-tracing.

Le volume balayé étant approché, on est en droit de s'interroger sur la qualité de la vérification que représente cette soustraction: permet-elle une comparaison efficace de la pièce de référence PC avec le brut usiné BU?

Diverses considérations entrent en jeu:

- les affichages simultanés de PR et BU permettent une première vérification visuelle sommaire. En affectant une couleur différente au support du brut ou à des parties qui

ne devraient pas être découvertes, on peut mettre en avant une mauvaise trajectoire d'outil mais ceci ne permet tout de même pas de mettre en évidence les différences entre les deux pièces.

- un moyen de remédier à cela est d'afficher les parties de PR qui ne sont pas dans BU et celles de BU qui ne sont pas dans PR, c'est-à-dire la différence symétrique de PR et de BU:

$$DS = (PR - BU) \cup (BU - PR)$$

Au mieux, DS est vide et les deux pièces sont égales.

- cette amélioration a encore le désavantage ne ne pas permettre une évaluation précise de l'erreur, même si on la situe plus facilement.
  On peut alors envisager une approche statistique calculant la proportion de PR que représente DS. C'est possible sans même augmenter sensiblement le nombre d'opérations puisque le calcul s'appuie sur des rayons qu'il faut de toute façon lancer pour l'affichage. Mais quoi que l'on fasse, on n'aura toujours pas d'informations précises sur une erreur donnée.
- on est donc amené à envisager le calcul des limites du brut, du brut usiné, de la pièce et de la différence symétrique de façon à pouvoir procéder à des interrogations plus précises (des cotations par exemple).
  Mais il est probable que là encore surviendront des difficultés du fait de la grande ressemblance des deux pièces sur lesquelles porte la différence symétrique (tangences, superposition ...).

2.4.5. Conclusion sur les enveloppes

Les enveloppes apparaissent comme un moyen de mettre en évidence un modèle mathématique exact d'un objet extrudé. On distingue dans le domaine:

- les approches au cas par cas, consistant à calculer manuellement les équations de l'enveloppe d'un type de mouvement et à en faire usage ensuite dans la programmation.
- les approches plus expertes (sans nuance péjorative pour celles qui précèdent) visant à un calcul automatique de l'enveloppe à partir de la définition implicite ou paramétrique du mouvement.

Dans le premier cas, on limite évidemment les possibilités du système au nombre de cas prévus et résolus par le développeur. Ceci dit, même avec cette approche, des problèmes considérables se posent encore, qui ne sont guère détaillés, même dans les publications s'y

rapportant: il s'agit par exemple de calculer la portion utile de l'enveloppe ou d'appliquer cette théorie à d'autres types de familles que les familles de plans, de sphères ou de cylindres.

Dans le deuxième cas, on fait intervenir des techniques de calcul formel qui doivent être suffisamment puissantes pour éliminer le paramètre t entre les deux équations qui caractérisent l'enveloppe et suffisamment robustes pour gérer les cas dégénérés (comme le cylindre qui dégénère en droite en 2.4.3.2.), raisonner sur l'intérêt de l'enveloppe calculée (nous avons vu dans l'exemple 2 du paragraphe 2.4.3.1. qu'une enveloppe peut exister même si la zone balayée par le générateur est infinie: elle ne sert donc ici à rien), les approximations...

On trouve dans [MAR 90] une discussion sur les techniques de calcul qui devraient permettre d'aboutir à l'équation implicite de l'enveloppe. Il cite:

- le changement de variable. L'un des plus connus est celui transformant une expression trigonométrique en une expression polynomiale:
  $\cos(v) = (1-u^2) / (1+u^2)$, $\sin(v) = 2.u / (1+u^2)$ et $u = tg(v / 2)$
- l'introduction d'équations supplémentaires. Ainsi, toujours pour éliminer des fonctions trigonométriques, on pose:
  $c = \cos(u)$, $s = \sin(u)$ et $s^2 + c^2 = 1$
- la méthode des résultantes, exploitée dans [SAH 90]
- le calcul des bases de Gröbner [BUC 88]

A l'heure actuelle, il semble, bien que des systèmes de calcul formel existent, que cette approche experte n'ait pas encore été mise en oeuvre.

L'approche plus traditionnelle, consistant à calculer manuellement les équations de l'enveloppe, semble, elle, bloquer davantage sur des problèmes de mise en oeuvre inhérents aux modèles de courbes et surfaces gauches que sur des aspects théoriques de l'enveloppe (c'est par exemple le cas pour le calcul des intersections d'une surface réglée avec une surface développable, enveloppe d'une famille de plans).

2.5. Conclusion sur la modelisation des objets extrudés

Ce chapitre était consacré à la modélisation d'objets extrudés définis par un couple (générateur,déplacement): nous nous sommes donc intéressés aux moyens de stocker de tels objets, en vérifiant, pour chaque structure étudiée, qu'elle autorisait également d'autres opérations que le stockage.

C'est ce critère qui nous a amené à déconseiller l'usage d'un modèle consistant à ne représenter les objets qu'à l'aide d'un couple (générateur,déplacement). Il est vrai que dans cette approche, il est simpliste de stocker un objet extrudé. Par contre, il semble impossible de combiner de telles entités; en effet, la nécessité de modéliser le résultat des opérations de combinaison (ou d'autres) implique de présenter ces derniers dans le format utilisé pour le stockage, à savoir un générateur et un déplacement: or, dans la majorité des cas, cette opération paraît éminemment complexe.

Nous avons donc, dans un deuxième temps, observé les possibilités offertes par les arbres de construction.

L'étude bibliographique a montré qu'ils se prétaient relativement bien au stockage et à la combinaison d'un type précis d'extrusions: les objets engendrés par le déplacement d'un contour plan (prismes, surfaces de révolution et cylindres généralisés). Il nous a semblé cohérent de faire apparaître ces objets aux feuilles des arbres, ce qui motive deux remarques:

- on étend alors de manière remarquable la puissance du schéma C.S.G. puisque la panoplie des volumes primitifs est considérablement plus variée.
- on a ainsi la possibilité de combiner entre eux ou avec d'autres types de volumes les objets extrudés, ce qui était impossible avec le modèle précédent.

Par contre, nous avons constaté que le schéma C.S.G., s'il permet facilement de représenter tout autre type d'extrusion, se prête difficilement à l'évaluation des nouveaux objets ainsi introduits, en particulier par des techniques de lancer de rayons (ray-casting).: il apparaît donc que cette approche reste encore insuffisante, en particulier dans les domaines de l'usinage, de la cinématique ou des mécanismes, où l'on s'intéresse aux mouvements de volumes.

En conséquence, nous avons abordé les modèles par les limites sous cet angle, en essayant de savoir s'ils se prétaient mieux au stockage d'objets définis par des volumes. Il est apparu qu'effectivement, grâce à l'apport des modèles mathématiques (surfaces gauches, surfaces réglées, surfaces enveloppes de familles de surfaces), il offraient davantage de possibilités dans ce domaine et qu'une bonne partie des difficultés soulevées était inhérente aux modèles surfaciques (intersection de surfaces gauches).

Ce chapitre ne se veut pas une étude exhaustive des techniques de modélisation d'objets extrudés. Nous avons cité des points que nous jugions importants, novateurs ou méconnus. Quand cela a été nécessaire, nous avons rappelé des notions anciennes ou plus connues.

Pour une vision plus complète, on pourra encore consulter:

- [CHO 90] (déjà cité en 1.2.2.3.) où l'on étend des notions anciennes comme les surfaces réglées, les surfaces de révolution ou les surfaces de raccordement entre des sections de passage en y intégrant des déformations du générateur ou des sections.
- [COQ 87b] où l'on donne une représentation paramétrique B-spline de la surface balayée par un contour, lui-même étant une courbe B-spline plane. La encore, le contour se déplace perpendiculairement à la trajectoire (cylindre généralisé), mais il peut être déformé par un changement d'échelle.
- [FAU 79] où l'on consacre un chapitre entier aux surfaces s'appuyant sur des sections de passage toutes perpendiculaires à un axe donné.
- [SHI 82] pour une formalisation de la modélisation d'objets extrudés dans un arbre C.S.G.

CHAPITRE III

3. - IMPLANTATION

Nous nous interessons ici à la mise en oeuvre de deux algorithmes d'extrusion que nous avons intégré au système SACADO: le prisme généralisé et l'objet de révolution, le premier parce qu'il autorise la construction d'objets variés avec un dialogue simple, le deuxième parce qu'il est l'occasion d'une discussion des opérateurs d'Euler.

Le modèle utilisé pour l'implantation est un B-rep polyédrique classique faces-contours-arêtes-sommets dont la description est donnée dans la première annexe. Les algorithmes cités ci-dessous s'attacheront donc à évaluer une approximation par des facettes planes ou quasi planes de la frontière des objets construits.

Dans le principe, ce sont deux méthodes permettant de définir des volumes à l'aide de contours, donc de faire le lien du 2D vers le 3D. Nous verrons à la fin de ce chapitre qu'elles se prêtent de fait à la production de pièces paramétrées.

3.1. Le prisme généralisé

Nous appelons prisme généralisé un volume construit par une opération simulant le collage d'un certain nombre de volumes élémentaires adjacents, définis chacun par un couple de contours et appelés abusivement cylindres (fig 3.1).

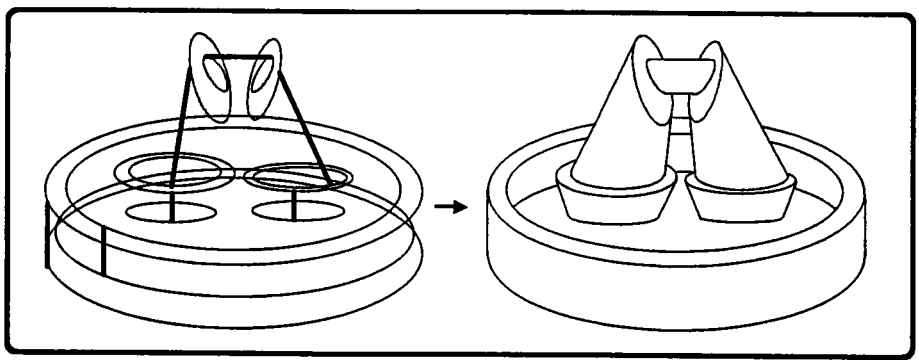


figure 3.1: les couples de contours et l'objet engendré

Dans la syntaxe que nous avons adoptée, les contours doivent être situés de manière à ce que les collages se fassent par les bases (il y a deux bases par cylindre), sans porte-à-faux ni intersection d'aucune sorte entre les contours bordant les bases (fig 3.2). On a adopté la convention que le fait de coller un cylindre à l'intérieur d'un autre créait du vide.

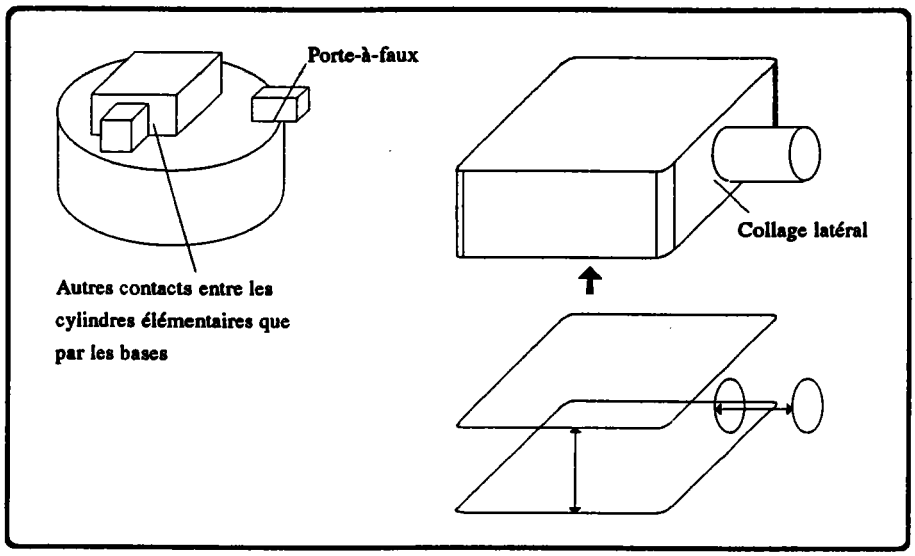


figure 3.2: les configurations interdites

3.1.1. Historique et intérêt d'un tel opérateur

A l'origine, les spécifications du prisme généralisé étaient plus restreintes: les volumes élémentaires étaient des prismes, aux bases toutes parallèles et aux arêtes colinéaires (fig 3.3)

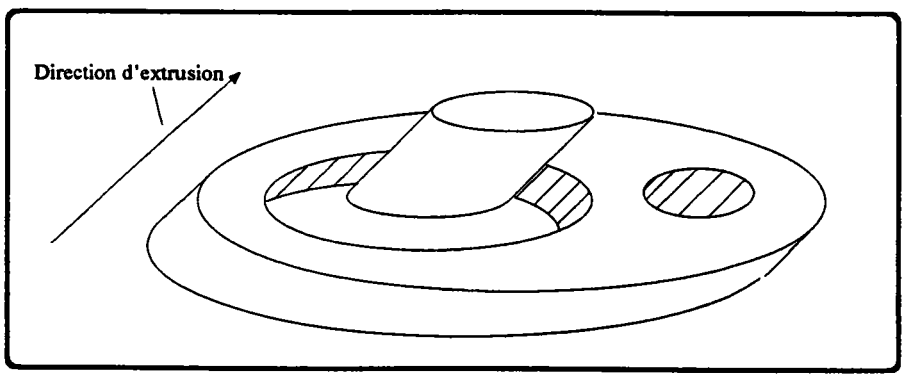


figure 3.3: première syntaxe du prisme généralisé

Ils étaient donc complètement définis par la donnée d'un contour de (xOy) et de deux "hauteurs" (en fait deux abscisses sur la droite orientée définissant la direction d'extrusion).

Ce type de syntaxe a deux avantages:

- le dialogue est tout-à-fait simple puisqu'il consiste essentiellement à fournir des informations 2D:
  - . pour la construction des contours, qui se pratique avec la version 2D du logiciel utilisé pour l'implantation.

. pour la donnée des deux hauteurs sur la direction d'extrusion

Avec un dialogue reposant sur des notions ensemblistes (union et différence), la tâche de l'utilisateur est plus contraignante puisqu'il doit créer les prismes élémentaires, les translater jusqu'au bon niveau puis demander une opération d'union ou de différence.

- les calculs sont simples et robustes, ce qui ne serait probablement pas le cas si on utilisait une approche ensembliste, notamment à cause des problèmes d'adjacence entre les volumes élémentaires.

Tel que, le prisme généralisé se pose donc comme une alternative aux techniques - interactives et calculatoires - dites par arbre de construction.

Par la suite, on a souhaité pouvoir relier entre elles des faces non parallèles, non identiques et sans toujours utiliser la même direction d'extrusion, ce qui a mené à la syntaxe actuelle. Le dialogue s'en est trouvé compliqué et ne présente plus guère d'avantages par rapport à celui qu'il faudrait mettre en place pour créer les volumes élémentaires, les situer dans l'espace et les combiner par opérations booléennes.

Les calculs sont cependant encore très simplifiés.

3.1.2. Algorithme

Contrairement à ce que laisse entendre la définition ci-dessus, la méthode ne consiste pas à créer un certain nombre de cylindres élémentaires, ensuite réunis par des opérations de collage; ceci conduirait:

- dans le cas où l'objet élémentaire définit du vide, car collé à l'intérieur d'un autre, à inverser le sens de parcours de ses faces, de façon à rester cohérent avec la convention de la normale extérieure.
- à transformer les contours extérieurs de certaines bases en contours intérieurs d'autres faces (aux endroits des collages).

On cherchera plutôt à déterminer à l'avance l'orientation de chaque contour de manière à créer des faces trouées immédiatement cohérentes aux lieux des collages (sens du contour extérieur opposé à ceux des contours intérieurs) et à tendre des bandes de facettes, entre les deux membres d'un couple de contour, elles aussi correctement orientées.

Les informations nécessaires à ces calculs sont presque en totalité présentes d'une manière implicite dans les données, à savoir une liste de couples de contours.

Nous nous attachons donc dans les lignes qui suivent, en particulier au travers de quelques propriétés des prismes généralisés, à montrer comment extraire ces informations.

Définition : *Deux contours appariés sont dits **analogues** l'un de l'autre ou en **vis-à-vis**.*

Propriété 1: *Connaissant l'orientation d'un contour C, on peut calculer celle de son analogue C', sauf si la droite d'intersection entre les plans des deux contours coupe C ou C'.*

Un de nos objectifs est de déterminer, avant la création effective de faces dans le modèle B-rep, quels seront, dans le prisme généralisé, les sens de parcours des contours passés en argument.

En cela, la propriété 1 est utile puisqu'elle permet de déduire de l'orientation d'un contour (orientation=sens de parcours) celle de son analogue (le contour auquel il est apparié). On la justifie ainsi:

que ce soit sur un cylindre, un prisme ou n'importe quel volume obtenu par raccordement de deux faces parallèles, les deux faces en question ont des sens de parcours contraires, autrement dit, le produit scalaire de leurs normales est négatif. Connaissant le sens de parcours de l'une d'elles, on peut donc automatiquement fixer celui de l'autre d'une manière cohérente.

Quand les plans des deux faces ne sont plus parallèles, cette règle doit être légèrement complétée: en fait, on remarque que ce n'est pas tant l'orientation des normales l'une par rapport à l'autre qui importe, mais plutôt l'orientation des normales vis-à-vis du corps du cylindre: voyons cela sur la figure 3.4.a; on cherche à calculer la normale N2 du deuxième contour: il faut donc choisir entre U et V en s'aidant de N1 qui est connue. La règle du produit scalaire (négatif) nous amène à choisir U, pourtant il est clair (figure 3.4.b) que le bon choix est V.

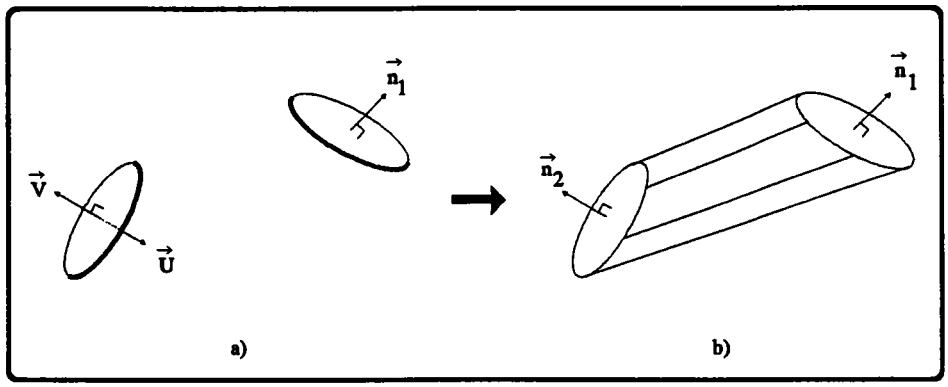


figure 3.4

En fait, le bon critère est ici de comparer l'orientation des normales vis-à-vis du corps du cylindre: si N1 "va vers le corps du cylindre", alors N2 doit faire de même et vice versa. Ceci peut être formalisé aisément ainsi:

Soient A et B deux points quelconques de C1 et C2, deux contours appariés,
alors
l'orientation de N2 doit être telle que:

$$\text{signe}(N1 \times AB) = \text{signe}(N2 \times BA)$$

On est alors amené à envisager des variations du signe de ces produits scalaires et, effectivement, lorsque l'un des contours est coupé par la droite d'intersection entre les deux plans, on constate qu'un des produits change de signe (fig 3.5.a) (on suppose que les deux contours ne sont pas coupés par la droite car un raccord valide est alors impossible).

Nous appellerons ce genre de situation un cas *bloquant* ou *équivoque*; en effet, la règle 1 ne permet plus de faire un choix et ce pour une très bonne raison: il y a deux cylindres possibles (fig 3.5.b).

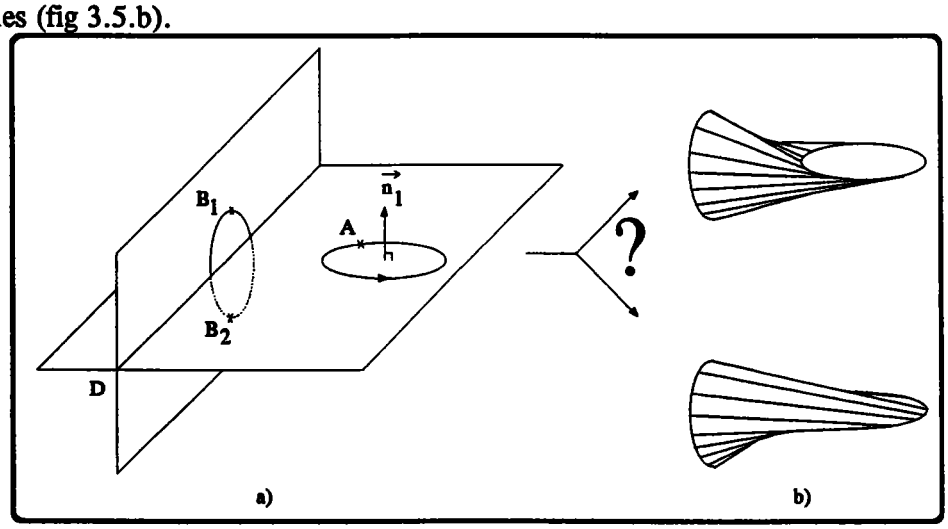


figure 3.5: le produit N1 x AB n'a pas le même signe suivant que B est d'un coté ou de l'autre de D.

Par conséquent, que l'on connaisse le sens de l'un ou de l'autre contour, il y a deux orientations possibles pour l'analogue; cette situation de blocage, si la suite de l'algorithme n'y apporte pas solution (on verra que les choses peuvent s'arranger d'elles-mêmes), sera traitée avec l'aide de l'opérateur: après avoir affiché les deux cylindres possibles, on lui demandera lequel est souhaité et c'est alors seulement que l'on pourra, connaissant le sens de parcours d'un des contours, déduire celui de son vis-à-vis (voir pour cela le corollaire de la quatrième propriété).

Propriété 2: *De tous les contours passés en argument, il y en a un au moins dont on peut immédiatement trouver l'orientation.*

Parmi les couples de contours donnés, certains définissent des cylindres de matière, d'autres de vide. A priori, il n'est pas simple de les distinguer, à quelques exceptions près. Le contour contenant le point de plus forte cote (et d'abscisse maximum s'ils sont plusieurs et d'ordonnée maximum s'ils sont encore plusieurs) est forcément la base d'un cylindre de matière. Les sens de parcours des deux bases sont donc tels que les normales sortent du cylindre, ce qui peut être également exprimé ainsi: la normale à une base doit s'éloigner de l'autre base ou, plus formellement:

si (C1,C2) est le couple de contours contenant le point extrême,
si A est un point de C1, B un point de C2,
alors les normales N1 et N2 de C1 et C2 doivent être telles que

$$N1 \times AB < 0 \text{ et } N2 \times BA < 0.$$

Dans la majorité des cas, ceci permet de déduire le sens de parcours des deux contours concernés. Toutefois, quand leurs plans sont sécants et que la droite d'intersection coupe un contour (il s'agit du même cas problématique que ci-dessus), l'orientation de l'un d'entre eux reste incertaine. Mais, et c'est ce qui servira dans l'algorithme, le deuxième, celui qui est coupé par la droite, est immédiatement orientable: sa normale doit s'éloigner de son analogue.

Dans tous les cas, il est donc possible de fixer d'une manière certaine le sens de parcours de l'un des contours passés en argument.

Propriété 3: *Connaissant l'orientation d'un contour C, on peut calculer celle des contours du même plan participant à la même face que C.*

D'après la définition que l'on a donnée du prisme généralisé, deux contours doivent être coplanaires et s'inclurent l'un l'autre pour que les deux volumes qu'ils engendrent avec leurs analogues respectifs, soient réunis en un seul.

On peut donc recenser les contours par plans. Considérons un de ces plans: il est porteur d'un certain nombre de contours. En étudiant leurs inclusions à l'aide d'un algorithme simple donné en annexe 2, on peut les ordonner et établir les faces mêmes qui apparaîtront dans le volume final (fig 3.6).

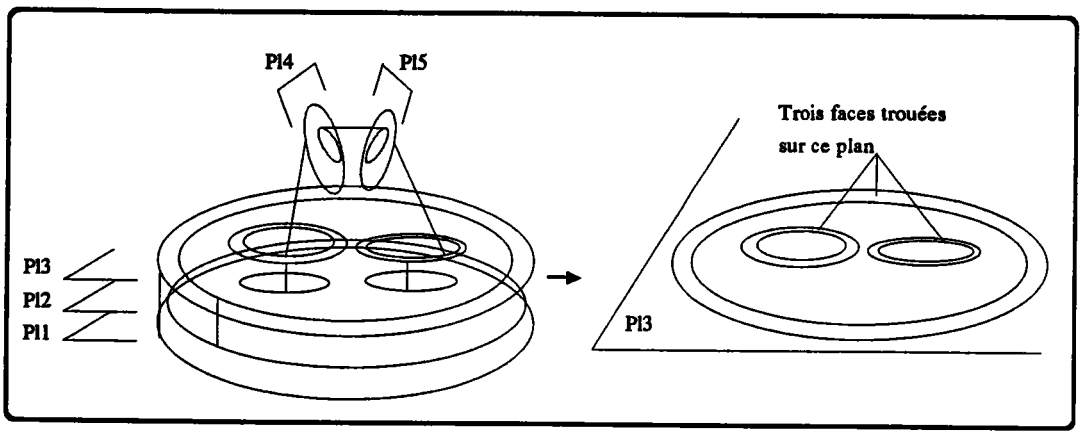


figure 3.6: détail d'un des plans porteur de contours: un calcul des imbrications permet de dégager immédiatement les trois faces qui apparaîtront à ce niveau.

Ces dernières n'ont alors qu'un défaut, celui de ne pas être correctement orientées, mais au moins sont elles bien limitées. A supposer alors que l'on détermine, pour chacune d'elles, l'orientation d'un seul de ses contours, on pourra, puisque les trous tournent à l'envers de l'englobant, supprimer ce dernier défaut et obtenir des faces tout-à-fait correctes.

Propriété 4: *Connaissant l'orientation d'un contour C et celle de son analogue C', on tend facilement sur (C,C') une bande de facettes dites 'latérales' correctement orientées en regard du volume global à construire.*

Le deuxième objectif de cet algorithme était de produire des facettes 'latérales' de raccordement d'orientation immédiatement correcte en regard du volume global. Ceci se réalise facilement si les deux lignes fermées (directrices) à raccorder sont elles-mêmes déjà correctement orientées: une arête devant être parcourue une fois dans chaque sens, il n'y a qu'un choix possible pour l'orientation des facettes de raccordement (fig 3.7).

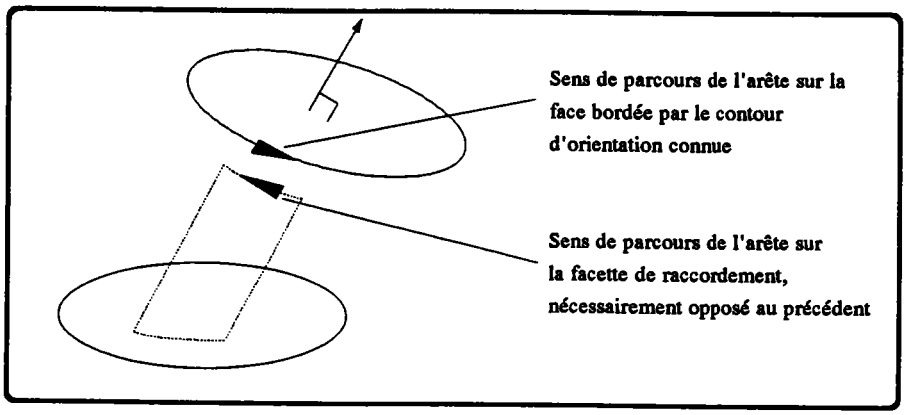


figure 3.7

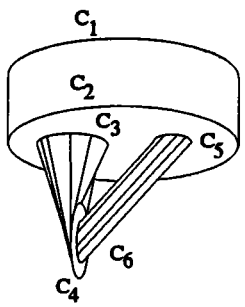
On déduit d'une manière immédiate de la quatrième propriété le corollaire évoqué et utilisé pour la première propriété:

Corollaire: *Connaissant l'orientation d'un contour C et celui des facettes latérales du cylindre qu'il engendre avec son analogue C', on détermine facilement l'orientation de C'.*

Partant de ces quatre propriétés, l'algorithme s'énonce alors ainsi: on peut, d'après la deuxième propriété, orienter l'un des contours d'une manière sûre et, d'après la troisième, en déduire le sens de rotation des autres contours de la face à laquelle il participe. Ceux-ci ont des analogues qui, dans la majorité des cas, peuvent à leur tour être orientés grâce à la première propriété.

On réitère alors le processus en cascade, orientant les faces de ces analogues puis les analogues des contours de ces faces...

Certains filets de la cascade ont pu aboutir à des impasses, à cause de la limitation de la première propriété (cas bloquants). On fait alors intervenir l'opérateur pour qu'il précise ses intentions, puisque l'on ne sait pas, à priori, quelle solution choisir. Cette intervention de l'opérateur peut survenir dès l'arrivée sur un cas bloquant ou après la fin de la chaine des orientations. Après réflexion, il apparaît que la deuxième solution est la meilleure car il se peut, dans la version actuelle (fig 3.8) et d'autant plus quand la première propriété aura été améliorée (voir paragraphe suivant), que la suite des orientations "revienne" sur le cylindre équivoque par son autre extrémité et qu'il soit alors possible de conclure: une intervention humaine est alors inutile.



Cascade des orientations:

. C1 et C2 sont orientés (propriété 2)
. on déduit de C2 les orientations de C3 et de C5
. tentative pour propager les orientations à partir de C3, mais c'est un cas bloquant ==> C4 n'est pas orienté
. déduit de C5 l'orientation de C6
. déduit de C6 l'orientation de C4

==> tous les contours sont orientés malgré le cas bloquant et sans intervention extérieure.

figure 3.8

Propriété 5: *Si, à l'issue de la cascade des orientations, certains contours n'ont pas été visités, c'est qu'un cylindre élémentaire au moins n'est adjacent à aucun autre et que, par conséquent, le volume à construire est constitué de plusieurs parties connexes. L'algorithme est alors relancé jusqu'à épuisement des contours.*

La totalité de l'algorithme est résumée ci-dessous dans un langage plus formel et des exemples d'exécutions (copies d'écran) sont fournis sur la planche 1 (en fin de paragraphe).

ALGORITHME DE CALCUL DE LA FRONTIERE D'UN PRISME GENERALISE

DONNEES : Une liste *LC* de contours appariés.

RESULTATS : Une liste *LV* des parties connexes de l'objet construit.

DEBUT

. Prétraitements (vérifications des arguments):
Pas de couples dont les deux contours sont coupés par la droite d'intersection de leurs plans.
Pas de couples dont les contours sont coplanaires.
. Sur chaque plan, reconstituer des faces informelles; les chainer dans une liste *lf_non_orientées*.
. (1)
. Créer deux listes vides des faces orientées (*lf_orientées*) et orientables (*lf_orientables*).
. Orienter un des contours (prop 2)
transférer sa face de *lf_non_orientées* vers *lf_orientables*.

{ orientation en cascade des contours d'une partie connexe }

tant que (*lf_orientables* est non vide) **ou**
(une face de *lf_non_orientées* est marquée)
faire

si (*lf_orientables* est vide) **alors**
{ la cascade est épuisée: l'opérateur intervient pour choisir entre les deux volumes possibles s'appuyant sur le contour bloquant de la première face marquée; les faces marquées parce que contenant un contour bloquant, l'ont été aux itérations précédentes }
F = première face marquée non orientée; C = contour bloquant de F
Choix utilisateur entre les deux cylindres possibles affichés.
Déduire l'orientation de C à partir du cylindre choisi.
Oter la marque de F.
Transférer F de *lf_orientees* vers *lf_orientables*.
fsi

F = tête (*lf_orientables*).
Orienter définitivement F à l'aide de son contour sûr (prop 3).

pour chaque contour C de F **faire**
Retirer C de LC.
si (l'analogue C' de C participe à une face non orientée) **alors**
orienter C' (prop 1).
si succès **alors** **{ cas non bloquant }**
F' = face comprenant C'.
démarquer F' si elle est marquée.
transférer F' de *lf_non_orientées* vers *lf_orientables*.
sinon **{ cas bloquant }**
marquer F' et C'.
fsi
fsi
fpour

Transférer F de *lf_orientables* vers *lf_orientées*.
ftq

{ création d'une liste lf_latérales des faces latérales, correctement orientées }

. *lf_latérales* = liste vide

. **pour** chaque face *F* de *lf_orientées* **faire**

pour chaque contour non marqué *C* de *F* **faire**

{un contour est marqué si son cylindre est déjà créé}

Soit *C'* l'analogue de *C:* marquer *C* et *C'*.

Abaisser des facettes correctement orientées sur (*C*,*C'*) et les ajouter à *lf_latérales*.

fpour

fpour

. Ajouter à *LV* le volume constitué des faces de *lf_orientées* et *lf_latérales*.

. **si** (*lf_non_orientees* est non vide) **alors**

{ volume constitué de plusieurs parties connexes }

Aller en (1)

fsi

FIN

3.1.3. Améliorations

Nous avons vu que la première propriété, permettant de déduire de l'orientation d'un contour C celle de son analogue C', n'est utilisable que si la droite d'intersection des plans des contours ne coupe ni C ni C'.

Dans le cas contraire et à moins d'un coup de chance, l'attitude adoptée est actuellement de faire intervenir un opérateur, ce qui n'est une bonne solution que si un choix automatique est vraiment impossible; et encore cela limite-t-il les possibilités de réévaluation automatique de la pièce (il faut conserver le choix de la personne dans l'historique de la pièce) ou de conception par langage de programmation (sans phase interactive). Nous proposons donc quelques moyens s'appuyant essentiellement sur des considérations géométriques d'améliorer les possibilités de la première propriété.

Rappelons d'abord que, dans les cas bloquants, il n'est pas possible de déduire l'orientation d'un contour de celle, connue, de son analogue parce que deux cylindres peuvent s'appuyer sur eux. Il vient alors à l'esprit d'essayer, au hasard, une des deux possibilités et de voir si elle mêne à une incohérence. Sur la figure 3.9, nous voyons une telle situation: un des choix amène les faces de l'objet à se recouper en d'autres lieux qu'en

leurs arêtes communes; l'autre possibilité est donc la bonne (à supposer que les arguments sont cohérents et qu'une construction est possible)

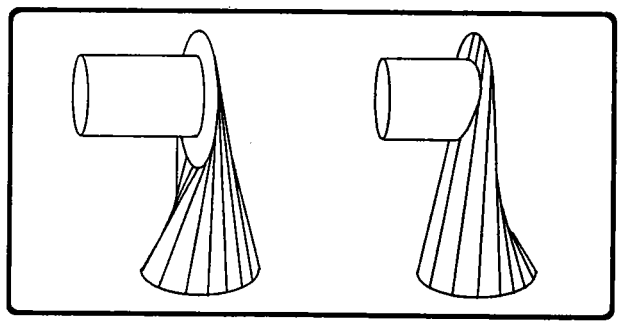


figure 3.9

Cette solution pouvant imposer des marches arrières, il est préférable de ne l'employer qu'en dernier ressort.

Une autre possibilité consiste, lorsque la cascade des orientations est bloquée, à relancer une ou plusieurs autres cascades: il suffit pour cela de rechercher un autre couple initial de contours, par exemple celui portant le point de plus forte abscisse ou, plus généralement, le point le plus éloigné dans une direction donnée. Il se peut alors que la deuxième série d'orientations rejoigne et débloque la première.

On peut enfin trouver une solution aux cas bloquants dans des critères radicalement différents, dépendant de l'application, comme par exemple l'exigence de maintenir à travers le cylindre équivoque le plus gros débit possible.

Ceci concernait les améliorations à apporter à la première propriété. L'autre défaut de l'algorithme dans sa version actuelle est que le maintien de la validité du résultat est entièrement à charge du concepteur: rien n'empêche par exemple qu'un trou ne descende plus profond que le cylindre qu'il perce (fig 3.10).

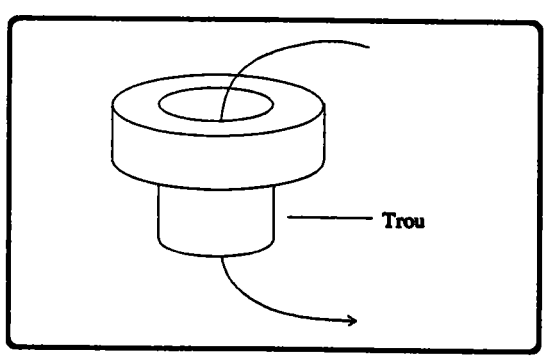


figure 3.10

Un test grossier pour détecter ce genre d'incohérence consiste à vérifier que les seuls lieux d'intersection entre les faces sont leurs arêtes communes mais il exige que soit terminée la création du volume: il serait préférable de savoir vérifier à priori la validité géométrique de l'objet.

Ceci mérite de plus amples réflexions que ne le suggère l'exemple du trou; il n'y a en effet rien de gênant à ce que le trou descende sous l'objet dès l'instant que celui-ci repose sur de la matière. On peut au passage remarquer que ces problèmes ne se poseraient pas d'une manière aussi aiguë avec une représentation C.S.G.

3.1.4. Extensions

En plus des améliorations évoquées ci-dessus, diverses extensions doivent être incorporées à la méthode, de manière à accroître la portée de l'opérateur. On peut citer:

- l'élévation à des hauteurs différentes de certaines zones d'une même face (pour l'instant, ces zones doivent être des contours intérieurs tous disjoints) (fig 3.11.a).
- la possibilité de lier un contour non plus à un mais à deux autres contours, de manière à intégrer la notion de tuyau évolutif (fig 3.11.b).
- des possibilités plus vastes au niveau du collage, par exemple:
  - . en autorisant un contact sur les faces latérales
  - . en autorisant les porte-à-faux; les faces apparaissant alors au niveau du collage seront le résultat d'opérations de différence entre les bases des cylindres élémentaires mis en cause (fig 3.11.c). A cette occasion, une réflexion a été menée sur les opérations booléennes entre contours et fait l'objet du paragraphe 3.2.
- un moyen de lier les contours par d'autres raccord que des surfaces réglées, par exemple en imposant que le début et la fin du cylindre soient perpendiculaires aux bases.

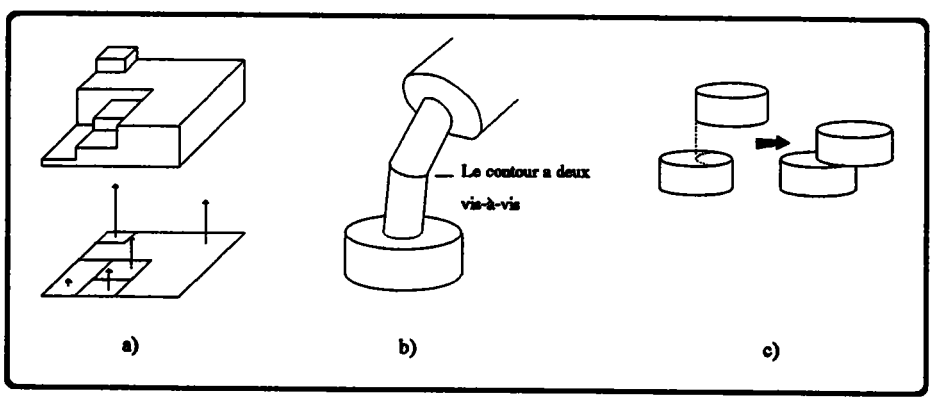
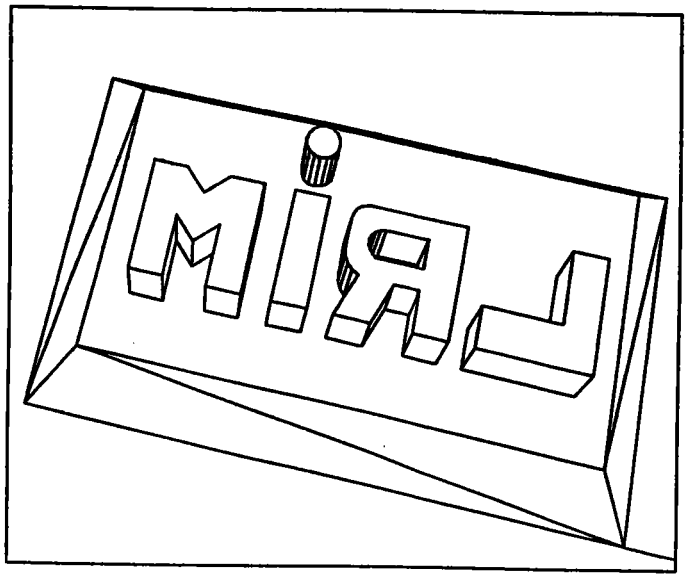
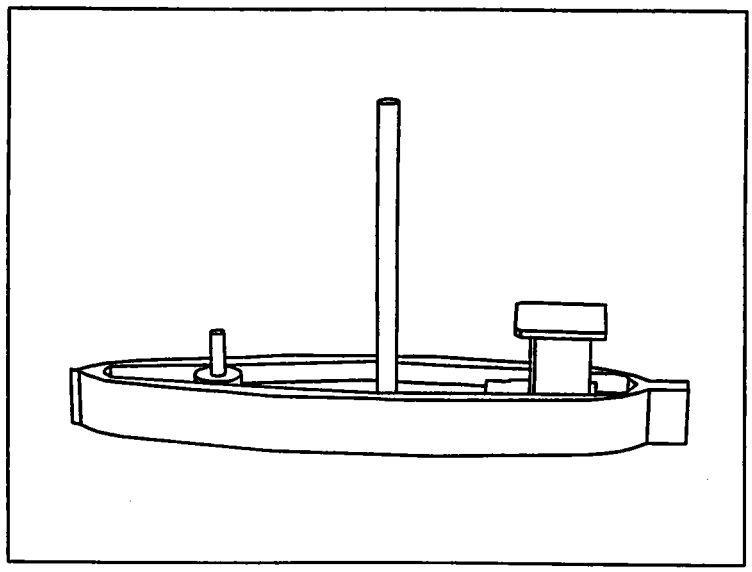


figure 3.11





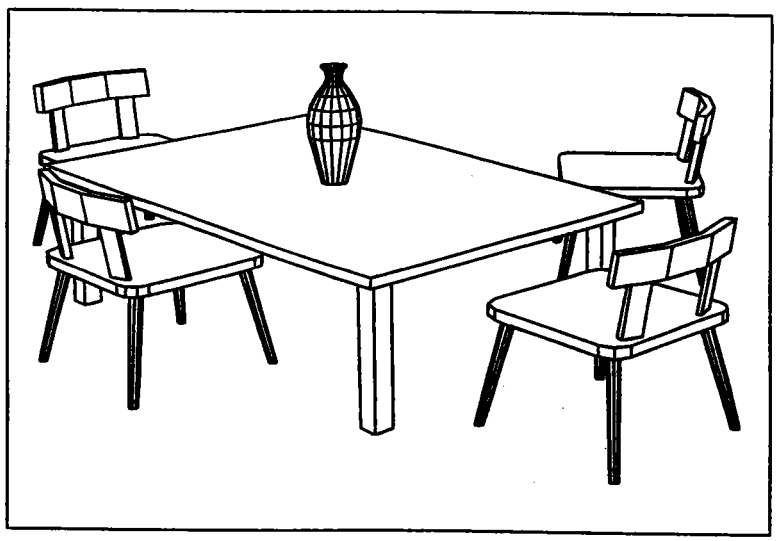


PLANCHE 1: Le tampon encreur, le bateau, la table et les chaises sont des prismes généralisés.

Le vase est un objet de révolution (voir paragraphe 3.3).

3.2. Opérations booléennes sur des contours

Nous évoquons dans ce paragraphe les problèmes posés par les opérations booléennes sur des contours, à savoir, pour deux contours coplanaires, le calcul de leur union, de leur différence ou de leur intersection.

Evidemment, un algorithme, très simple et très connu, existe pour ce genre de calculs: sans entrer encore dans le détail, disons qu'il consiste à partir d'un point que l'on sait appartenir au résultat, à progresser sur le contour portant ce point jusqu'à rencontrer une intersection et à reprendre la progression sur le contour croisé.

Ce cheminement se poursuit jusqu'à ce que l'on revienne au point de départ.

Les sens de progression sont liés aux sens de parcours (*supposés identiques*) des contours: dans le cas d'opérations d'union ou d'intersection, on progressera toujours selon les sens de définition des contours; dans le troisième cas, on empruntera toujours le contour qui est soustrait à contre-sens (fig 3.12).

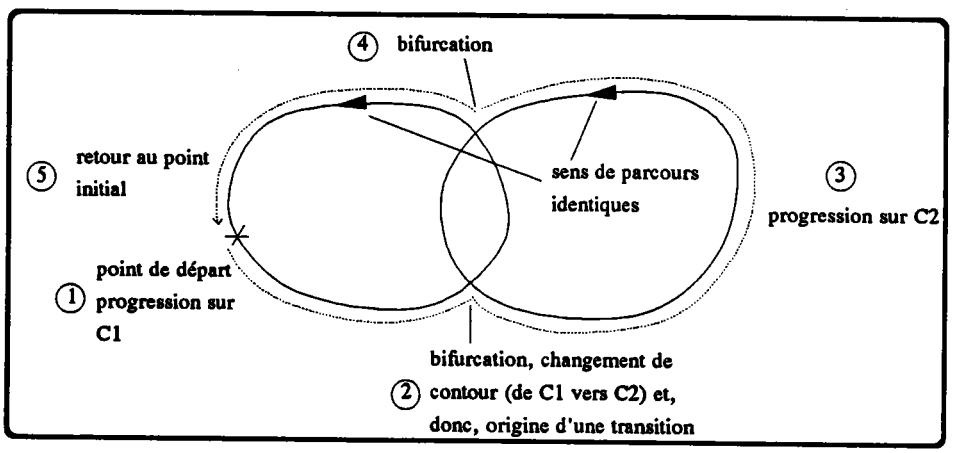


figure 3.12: calcul de C1 + C2

Cette brève explication suffit à poser les problèmes que l'on rencontre lors d'une mise en oeuvre et qui méritent que l'on s'y attarde puisqu'il est très difficile d'y trouver des allusions sous cette forme dans la littérature: on trouve en effet beaucoup d'allusions à la classification d'une droite par rapport à un polygone ou un contour [VAN 82]. Le plus souvent, ces problèmes sont rattachés à celui du clipping mais une application à la combinaison de contours apparaît rarement. Les difficultés caractéristiques de cette approche sont les suivantes:

- toutes les intersections entre les contours correspondent-elles à des bifurcations? (en particulier celles où il n'y a pas pénétration d'un contour dans l'autre et que nous appelons *contacts*)

- que faire lorsqu'une intersection n'est pas réduite à un point (les contours sont *adjacents* ou localement superposés)?

Nous reprenons ces points dans le paragraphe suivant dans lequel nous essayons de préciser quelle attitude adopter face aux divers types d'intersections rencontrées (traversées, contacts et adjacences) dans le cheminement et quand une telle intersection doit être considérée comme une bifurcation.

3.2.1. Préliminaires

Avant de chercher à savoir à quelles intersections bifurquer, nous reprenons l'algorithme dont les grandes lignes ont été énoncées ci-dessus afin de préciser quelques termes utiles dans la suite. Nous supposerons pour l'instant que toute intersection est une bifurcation.

En considérant la figure 3.13, cherchons à déterminer C1 + C2.

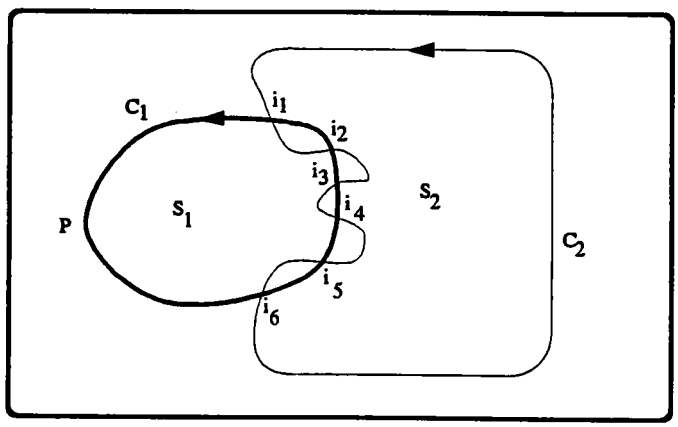


figure 3.13

Supposons que l'on parte de P. En vertu de ce qui précède, nous allons reconstituer le contour extérieur bordant S1 + S2 (les surfaces bordées par C1 et C2). Pour cela, nous aurons bifurqué en I6 et I1.

S1 + S2 est toutefois limité intérieurement par deux autres contours qu'il convient également de reconstituer. Il faut donc retrouver un point initial pour chacune de ces reconstructions.

Pour cela, plutôt que de chercher explicitement un point de C1 n'appartenant ni à S2 ni à la portion de C1 déjà utilisée pour le premier contour, nous partons d'une des bifurcations qui, lors de la reconstruction précédente, ont fait passer de C1 sur C2 (I6 est la seule possibilité sur notre exemple) et progressons sur C1 jusqu'à l'intersection suivante (I5). La portion de C1 ainsi parcourue n'est pas utile mais elle mène à un point qui se

révèle être soit le prochain point initial, soit un point appartenant déjà à un contour reconstitué. Dans le premier cas, on pourra alors relancer l'algorithme de reconstruction en avançant sur C1 à partir du point trouvé. Dans le deuxième cas, il faudra lancer une nouvelle recherche à partir d'une autre bifurcation (toujours parmi celles qui ont fait passer de C1 vers C2 dans une reconstruction précédente).

Définition: *Nous convenons d'appeler **suivi** la phase de reconstruction d'un contour et **transition** la phase consistant à parcourir une portion a priori inutile de C1 en vue de trouver un point initial pour un prochain suivi.*

L'algorithme se présente donc à présent comme suit:

- recherche d'un premier point initial sur C1 (nous traitons ceci en 3.2.5.)

(1)
- reconstruction d'un contour dans une étape de suivi
- transition à partir d'une bifurcation utilisée dans un suivi précédent pour passer de C1 vers C2.
  Si celle-ci ne mène pas à un nouveau point initial, on essaie avec une autre, puis une autre etc...
- deux issues sont alors possibles: soit on trouve un point initial et il faut relancer un suivi, donc retourner en (1), soit toutes les transitions ont échoué et le calcul est terminé.

Il peut cependant arriver que la situation soit suffisamment simple pour éviter une application complète de l'algorithme. Ces cas (triviaux) sont traités en 3.2.5.

Dans cette partie d'introduction, nous avons supposé, à tort, que toute intersection constituait une bifurcation ou marquait la fin d'une transition. Ceci ne remet cependant pas en cause l'algorithme qui précède dans lequel nous avons pris soin de parler de bifurcation et de fin de transition, non d'intersections: il convient simplement maintenant de préciser parmi toutes les intersections possibles celles qui sont effectivement des bifurcations ou des fins de transition. Celles qui n'en sont pas devront être négligées.

Pour l'essentiel, ces choix s'appuient sur les remarques suivantes.

Règle 1: Dans une opération d'union, puisque l'on cherche à entourer les points appartenant à S1 ou S2, il ne peut arriver, pendant le suivi, que l'on se trouve à l'intérieur d'un des contours. Durant une transition par contre, le cheminement se faisant sur C1, on est nécessairement à l'intérieur de C2.

Règle 2: Quand on calcule C1 & C2, les suivis doivent se faire à l'intérieur d'un contour; en effet, admettons par exemple que l'on progresse sur C1; si on suppose que l'on se trouve alors à l'extérieur de C2, c'est que l'on est en train d'entourer des points n'appartenant qu'à C1 et non pas à C1 & C2, comme souhaité. Lors d'un parcours de C1 (resp. C2), on doit donc impérativement se trouver à l'intérieur de C2 (resp. C1).
En phase de transition, par contre, puisque l'on progresse sur C1, on est à l'extérieur de C2.

Règle 3: Lorsque l'on calcule C1 - C2, on veut entourer les points de C1 n'appartenant pas à C2, ce qui empêche le suivi de pénétrer à l'intérieur de C2. Respectivement, pendant les transitions (qui se font toujours sur C1), on se trouve à l'intérieur de C2.

3.2.2. L'intersection est une traversée

Bien que nous l'ayons déjà employé, nous définissons clairement le terme de *traversée*.

Définition: *Un point d'intersection entre deux contours est une **traversée** si, en ce point, l'un des contours pénètre l'autre.*

Propriété: *Toute traversée rencontrée dans un suivi est une bifurcation; toute traversée rencontrée dans une transition marque la fin de la transition.*

Toutes les traversées sont donc significatives (dans la figure 3.12, toutes les intersections étaient des traversées). Celles qui, dans un suivi, auront fait passer de C1 sur C2 seront les points de départ d'étapes de transition.

3.2.3. L'intersection est un contact

Définition: *Un point d'intersection entre deux contours est un **contact** si, en ce point, il n'y a pas pénétration d'un contour dans l'autre. Localement, chaque contour est alors entièrement d'un même côté de l'autre.*

Des trois types d'intersection, les contacts sont les plus pénibles à gérer car ils amènent à distinguer un nombre relativement important de cas dont certains peuvent être résolus de deux manières différentes, ce qui implique un choix.

Nous sommes arrivés à une liste de 14 cas, fonction de:
- la nature de l'opération (+, &,-)
- la nature de la progression (suivi, transition)
- la position du contour rencontré vis à vis de celui sur lequel on se déplace (dedans, dehors).

Face à un contact, on réagira donc différemment suivant que l'on calcule une union, une intersection ou une différence, suivant que l'on se trouve en phase de suivi ou de transition et suivant le côté duquel se trouve le contour rencontré. Ces 14 cas sont résumés dans le tableau ci-dessous (fig 3.14).

Explications relatives à la table:

La position du contour rencontré (3^{e} colonne) est une information à caractère local: nous savons qu'au voisinage d'un contact, les contours restent d'un même côté l'un par rapport à l'autre: la position spécifie de quel côté - du côté matière ou extérieur - se trouve le contour rencontré vis à vis du contour sur lequel on est en train de cheminer et que l'on appelle le contour courant (colonne suivante du tableau).

Le contour courant change donc à chaque bifurcation et vaut alternativement C1 ou C2. Dans la colonne correspondante du tableau, une étoile signifie que la connaissance du contour courant est inutile. Deux étoiles signifient que le contour courant est forcément C1, car on est en phase de transition.

nature opération	nature progression	position contour rencontré	contour courant	attitude	longueur de la transition éventuelle	
∪	suivi	extérieur	*	N		1
		intérieur	*	B	nulle	
	transition	extérieur	* *	N		
		intérieur	* *	N		
∩	suivi	extérieur	*	B	nulle	5
		intérieur	*	N		
	transition	extérieur	* *	N		
		intérieur	* *	N		
C1 - C2	suivi	extérieur	C1	B	nulle	
			C2	N		10
		intérieur	C1	N		
			C2	B		
	transition	extérieur	* *	N		
		intérieur	* *	N		

figure 3.14: le tableau des 14 cas de contacts

Dans ces deux cas, l'information contour courant est donc sans importance: elle ne le devient que lors d'une opération de différence qui est la seule des trois à ne pas être commutative. L'avant-dernière colonne indique l'attitude à prendre face au contact rencontré: N est mis pour "à Négliger", B pour "Bifurcation". Parmi les 14 cas distingués, 10 sont à négliger donc ne constituent ni des bifurcations (lors d'un suivi) ni des fins de transition; seuls les quatre autres correspondent à des changements de contour. Rappelons que certains d'entre eux pourront provoquer un passage de C1 sur C2 et marqueront l'origine d'une transition. On indique alors une information, dans la dernière colonne relative à la transition, et expliquée dans l'exemple détaillé qui suit.

Nous ne justifierons qu'une ligne du tableau, les autres s'expliquant par des arguments similaires. Dans les illustrations, nous supposons que le sens de rotation des contours est tel que la matière soit à gauche; la matière est représentée par un léger hachurage d'un côté du contour.

Exemple: Justification des cinquième et sixième lignes du tableau

Situation: Au cours de la reconstruction de l'un des contours composant C1 & C2, on rencontre un contact.

Les rôles de C1 et de C2 étant tout-à-fait symétriques, on peut convenir que le contour actuellement emprunté par le suivi est C1 et que le contour rencontré est C2.

C2 est alors (localement) soit à l'intérieur soit à l'extérieur de C1 (fig 3.15.a et b).

En vertu de la règle 2, la portion de C1 sur laquelle on progresse est à l'intérieur de C2 ce qui implique que les sens de parcours de C2 sont ceux indiqués en (c) et (d).

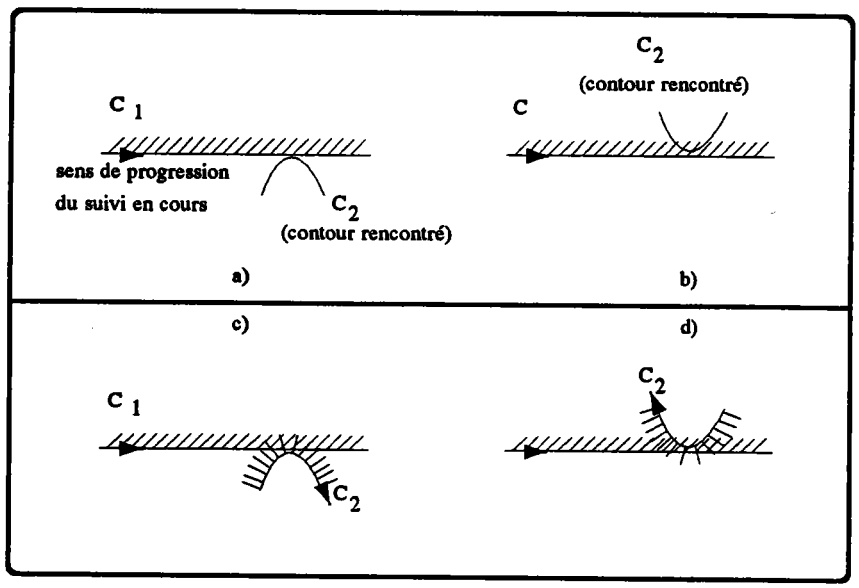


figure 3.15

On peut alors décider si les deux contacts possibles sont des bifurcations.

- si C2 est extérieur (c), il ne faut pas bifurquer car on ne se trouve plus à l'intérieur d'un contour ce qui va à l'encontre de la règle 2: le contact doit être négligé (ligne 6 du tableau)
- si C2 est intérieur, il y a matière à discussion; la figure 3.16.a donne une idée d'une telle situation

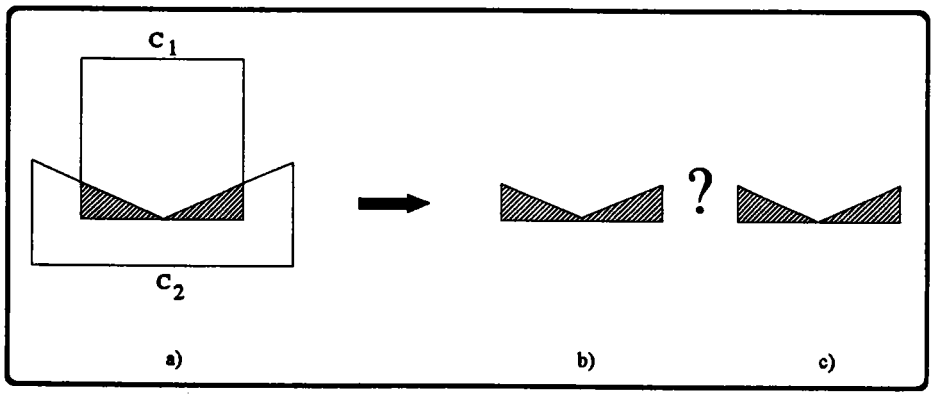


figure 3.16

On constate que C1 & C2 subit un étranglement au point de contact et deux résultats sont possibles (b et c): un contour se recoupant ou deux contours avec un point de contact (fig 3.16.b et c).

La première solution arrive lorsque l'on néglige le contact, la deuxième lorsque ce dernier est considéré comme une bifurcation.

Nous pensons qu'à quelques exceptions près correspondant aux cas triviaux, le deuxième choix est le meilleur: en effet nous avons fait l'hypothèse que C1 et C2 étaient valides, donc entre autre sans contact avec eux-même: or le premier choix entraîne l'existence d'un contour ne respectant plus cette convention, ce qui interdit d'en faire l'argument d'une autre combinaison booléenne. Ceci peut se révéler gênant en particulier lors de l'évaluation d'une expression ensembliste à plus d'un opérateur.
En effet, on évalue alors les expressions en partant de l'opération la plus imbriquée et on "remonte" petit à petit; or cette démarche est interdite si les opérations ne sont pas des lois de composition internes.

En conséquence, nous avons convenu que le cas (d) devait être considéré comme une bifurcation (ligne 5 du tableau).

Il pourra donc arriver, sur un contact, qu'un suivi passe de C1 sur C2: or nous avons dit que de telles bifurcations étaient les points de départ d'étapes de transition (recherche du point initial d'un prochain suivi). Observons alors quel aspect aura la transition en question. Pour cela, reportons nous à la figure 3.16.a. ci-dessus: il apparaît immédiatement qu'elle est de longueur nulle, autrement dit que le point initial du prochain suivi sera le contact lui même. C'est pour cette raison qu'il est indiqué une transition de longueur nulle dans la dernière colonne du tableau.

3.2.4. L'intersection est une adjacence

Définition: *Une **adjacence** est une intersection des contours non réduite à un point. Il s'agit donc d'une ligne commune à chacun d'eux. On distinguera les adjacences contact des adjacences traversées: dans le premier cas, les deux brins partant des extrémités de l'adjacence sur l'un des contours sont d'un même côté (intérieur ou extérieur) de l'autre contour. Dans le deuxième cas, les deux brins sont l'un à l'intérieur, l'autre à l'extérieur (fig 3.17).*

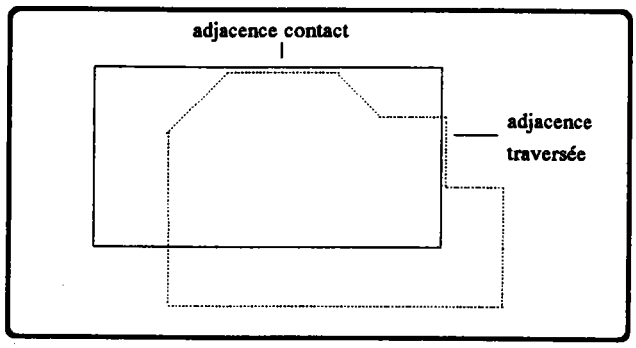


figure 3.17

Nous cherchons à présent si, lorsque le suivi rencontre une telle intersection, il faut bifurquer en début ou en fin d'adjacence ou, au contraire, ne pas en tenir compte.

Nous cherchons également à savoir sous quelles conditions une adjacence marque la fin d'une phase de transition.

A nouveau, le choix sera fonction:

- de la nature de l'opération (+,-,&)
- de la nature de la progression (suivi ou transition)
- des sens de parcours de la partie commune sur les deux contours (identiques ou opposés)
- de la nature de l'adjacence (traversée ou contact)
- du contour courant, dans le cas de la différence.

ce qui conduit à étudier 28 cas, résumés dans la table ci-dessous (fig 3.18); les conventions de notation y sont les suivantes:

- deux flèches dans le même sens indiquent que les sens de parcours d'une adjacence sur les deux contours qui la partagent sont identiques; deux flèches en sens inverses impliquent évidemment le contraire.
- les lettres T et C caractérisent respectivement les adjacences Traversées et Contact.

- la lettre B signifie Bifurcation: l'adjacence correspondante doit alors provoquer un changement de contour au cours du suivi.
  La lettre B est complétée d'un D ou d'un F suivant que la bifurcation doit se faire en début ou en fin de la ligne commune.
  Par convention, ce que l'on appelle *début* de la ligne commune est le premier point de l'adjacence rencontré par le suivi. Cette précision est utile lorsque l'adjacence est parcourue une fois dans chaque sens et que le début n'est plus une notion évidente.
  Le symbole FTFA tient pour "Fin de Transition en Fin d'Adjacence": il signifie que la fin de l'adjacence rencontrée au cours d'une transition marque aussi la fin de cette transition: la fin de l'adjacence constitue donc le début d'un prochain suivi, à moins que la portion de C1 partant de ce point n'ait déjà été utilisée.
  La lettre N signifie que l'adjacence peut être Négligée car elle n'a d'incidence ni sur le suivi, ni sur la transition en cours.
- lorsque l'adjacence entraîne un changement de contour (BD ou BF), on précise, pour le cas où la bifurcation se ferait de C1 vers C2, où commencer la transition correspondante et quelle sera sa durée: le début de la transition se situe toujours en fin d'adjacence (d'où le mot fin) mais il peut arriver que la transition soit de longueur nulle, autrement dit que la fin de l'adjacence soit le point initial d'un prochain suivi (à moins que la portion de C1, issue de ce point, n'ait déjà servi).

Pour clarifier les choses, voilà comment interpréter les 17è et 25è lignes du tableau:

Ligne 17:

Si, lors de la reconstruction d'un contour de C1 - C2, on arrive par C1 à une adjacence parcourue dans le même sens sur C1 et C2, et
Si cette adjacence est une traversée,
Alors passer sur C2 (et emprunter C2 à contre sens puisqu'il s'agit d'une différence) et noter que lorsque le contour en cours sera bouclé, on pourra chercher un point initial pour le contour suivant en lançant une transition sur C1 partant de l'extrémité de l'adjacence.

nature opération	nature progression	contour courant	sens parcours adjacence	traversée contact	attitude	origine et longueur de la transition	
∪	suivi	*	→→	T	BF	Fin Adj L ≠ 0	1
				C	N		
			→←	T	BD	Fin Adj L ≠ 0	
				C	BD	Fin Adj L = 0	
	transition	*	→→	T	FTFA		5
				C	N		
			→←	T	FTFA		
				C	N		
∩	suivi	*	→→	T	BF	Fin Adj L ≠ 0	9
				C	N		
			→←	T	BD	Fin Adj L = 0	
				C	BD	Fin Adj L ≠ 0	
	transition	*	→→	T	FTFA		
				C	N		
			→←	T	FTFA		
				C	N		
C1 - C2	suivi	C1	→→	T	BD	Fin Adj L ≠ 0	17
				C	BD	Fin Adj L = 0	
			→←	T	BF	Fin Adj L ≠ 0	
				C	N		
		C2	→→	T	BD		
				C	BD		
			→←	T	BF		
				C	N		
	transition	*	→→	T	FTFA		25
				C	N		
			→←	T	FTFA		
				C	N		

figure 3.18

Ligne 25:

Si, lors de la recherche du premier point d'un des contours issus de C1 - C2, on arrive à une adjacence parcourue une fois dans chaque sens et

Si celle-ci est une traversée

Alors la fin de l'adjacence est le point initial recherché, à moins que la portion de C1 issue de ce point ne soit déjà utilisée pour un autre contour de C1 - C2.

A nouveau, afin de ne pas trop alourdir la lecture, nous ne justifions qu'une des lignes du tableau.

Exemple: Justification de la ligne 19.

Les hypothèses de la 19è ligne correspondent à la figure 3.19.

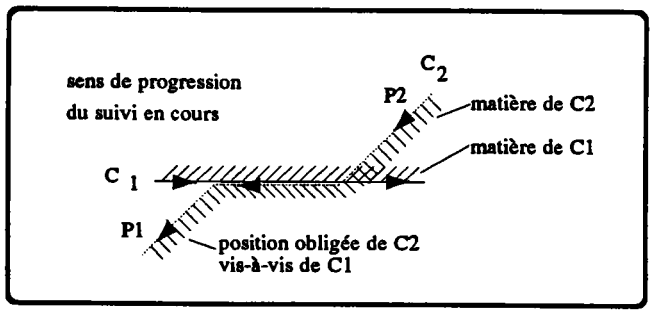


figure 3.19

En effet:

- on a vu que lors du calcul de C1 - C2, un suivi ne peut pénétrer à l'intérieur de C2 (règle 3): la portion P1 de C2 est donc impérativement à l'extérieur de C1.
- on a fait l'hypothèse que l'adjacence est une traversée, ce qui impose au brin P2 de C2 de se trouver à l'intérieur de C1.

On conclut alors facilement: pour continuer à entourer les points de C1 n'appartenant pas à C2, il faut nécessairement passer sur C2 et progresser le long de P2. On note qu'il est peu important de changer de contour en début ou en fin d'adjacence: nous avons arbitrairement choisi la deuxième solution.

3.2.5. Cas triviaux et premier point initial

Les cas triviaux surviennent lorsque les contours ne se coupent pas ou que leurs seules intersections sont des contacts (points ou adjacences). On peut alors parfois immédiatement présumer du résultat de leur combinaison, sans un recours à l'algorithme général dont les grandes lignes dont données dans les paragraphes précédents.

Une absence d'intersection implique que les contours sont disjoints ou que l'un est inclus dans l'autre; toute combinaison est alors simpliste (fig 3.20).

situation / opération	$C1 \subset C2$	$C1 \supset C2$	autres
$C1 \cup C2$	C2	C1	C1,C2
$C1 \cap C2$	C1	C2	$\emptyset$
C1 - C2	$\emptyset$	C1 troué par C2	C1
C2 - C1	C2 troué par C1	$\emptyset$	C2

figure 3.20

Si toutes les intersections sont des points ou des adjacences de contact, alors il est sûr que l'un des contours est entièrement d'un même coté de l'autre: en effet, cette propriété est vérifiée au voisinage des contacts et ne peut être enfreinte ailleurs puiqu'il n'y a pas d'autres intersections (comme des traversées, par exemple) qui permettraient à un contour de franchir l'autre.

Trois situations sont alors possibles:

- C1 est inclus dans C2 (fig 3.21.a)
- C2 est inclus dans C1 (fig 3.21.b)
- les intérieurs de C1 et de C2 sont disjoints (fig 3.21.c).

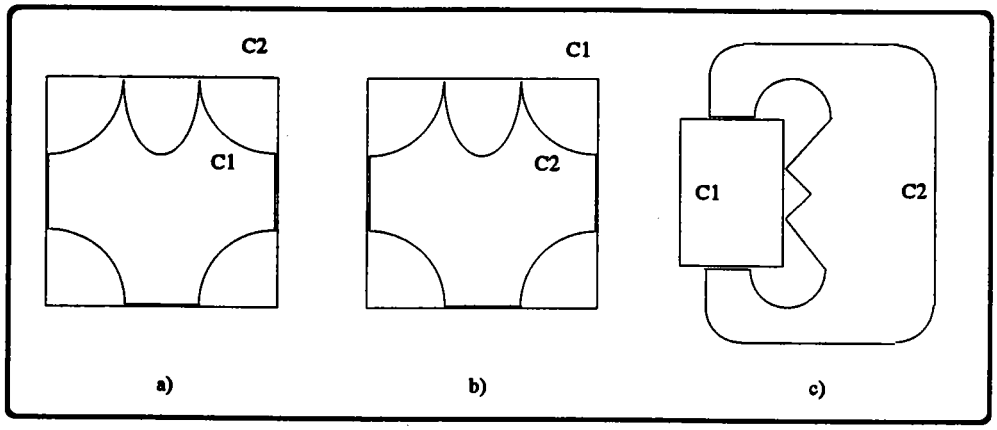


figure 3.21

Pour distinguer ces trois cas, il suffit de tester, par rapport à C2, la position d'un point de C1 qui ne soit ni un point d'intersection, ni sur une adjacence. Si le point est intérieur, alors C1 est inclus dans C2, sinon, il est complètement extérieur (aux intersections près). En faisant de même avec C2, on identifie facilement l'un des trois cas.

Si toutefois il se trouve au moins une adjacence dans la liste des intersections, il y a mieux à faire: en effet, le troisième cas se démarque du premier en ceci que les adjacences y sont toutes parcourues une fois dans chaque sens, alors que dans les deux autres cas, elles le sont toutes deux fois dans le même sens. Il suffit donc de tester le sens de parcours d'une quelconque des adjacences puis, si l'on se trouve dans l'un des deux cas d'inclusion, de ne tester l'intériorité que d'un seul point.

Pour chaque opérateur booléen, voyons maintenant quels cas triviaux amènent ces trois situations.

Pour l'union, (a) et (b) sont des cas triviaux (le résultat est l'englobant). Ce n'est par contre pas le cas pour (c), à moins que n'existe qu'une unique intersection et que celle-ci soit un point: nous choisissons alors ne pas employer l'algorithme, qui retournerait un contour avec une auto-intersection (contour en forme de "huit") mais plutôt d'agir comme si les contours étaient distincts.

Pour l'intersection, (a), (b) et (c) sont tous trois des cas triviaux: on n'utilisera donc pas l'algorithme et on rendra donc respectivement C1, C2 et l'ensemble vide (des contours).

Pour la différence, (a) et (c) sont triviaux et valent respectivement l'ensemble vide et C1. Par contre, le calcul de (b) impose l'emploi de l'algorithme, avec à nouveau une exception s'il n'y a qu'une intersection et que celle-ci est un contact: nous pensons alors plus adapté de proposer deux contours imbriqués partageant un point plutôt que le contour avec auto-intersection que nous rendrait l'algorithme (fig 3.22).

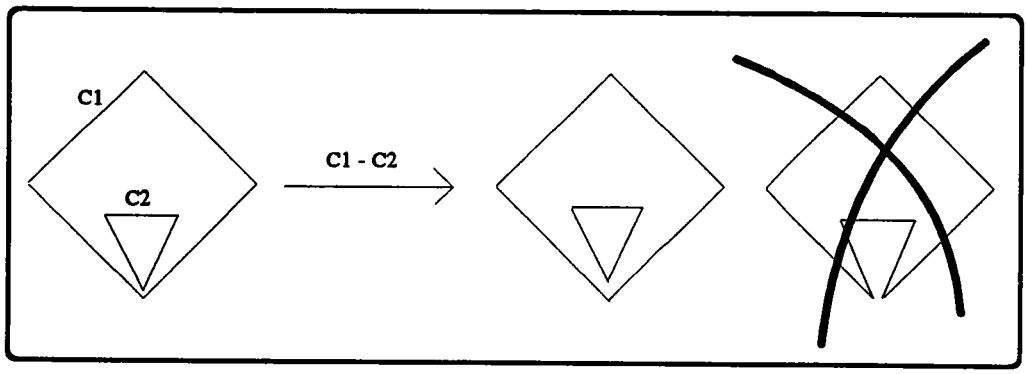


figure 3.22

Ces choix sont évidemment plus ou moins arbitraires: nous avons essayé de maintenir une certaine cohérence mais il est clair que l'on pourrait prendre d'autres décisions selon le contexte. En particulier, nous avons systématiquement évité de produire des contours se recoupant; si l'on préfère tolérer cet état de fait, un certain nombre de cas particuliers tomberont.

Nous terminons avec la recherche d'un point initial pour le premier suivi: on a vu qu'à l'aide d'étapes de transition, les points de départ des autres suivis peuvent être calculés automatiquement. Seul le premier d'entre eux nécessite donc une étude particulière.

Dans notre approche, un suivi démarre systématiquement sur C1, ce qui, après élimination des cas triviaux, est toujours possible. Il nous faut donc, en fonction de l'opération souhaitée, trouver un point de C1 qui soit:

- pour l'union, hors de C2
- pour l'intersection, à l'intérieur de C2
- pour la différence, hors de C2.

Ceci est réalisé simplement en parcourant C1 et en testant un point de chaque portion comprise entre deux intersections consécutives.

Si l'une des intersections au moins est une traversée, on optimise la recherche en vertu du fait que les brins de C1 qui précèdent et suivent l'intersection sont l'un dans et l'autre hors de C2. Il suffit donc de tester un point de l'un des brins: si celui-ci n'est pas satisfaisant, tout point de l'autre brin le sera.

3.2.6. Conclusion sur la combinaison de contours

Cette étude des opérations booléennes sur les contours nous a amené à distinguer une liste de cas relativement importante, que nous pensons exhaustive mais difficilement compressible.

les notions utilisées pour cela sont plutôt simples, bien que dans certains cas, nous ayons été amenés à faire des choix assez délicats; nous l'avons fait de la manière que nous jugions la plus cohérente mais au besoin d'autres options peuvent les remplacer: il suffit pour cela de corriger les valeurs des colonnes "attitude face à l'intersection" des tableaux des figures 3.14 et 3.18.

C'est pour cette raison que nous estimons inutile de chercher à compacter cette liste de cas qui, d'une part, n'est pas d'un volume extraordinaire et d'autre part, permet de modifier le comportement de l'algorithme en corrigeant simplement les tables qui le pilotent et sans intervention dans l'algorithme lui-même.

Nous n'avons effectivement programmé qu'une petite partie de ce qui précède, à savoir l'opération d'union sur des contours sans adjacences.

Ceci a cependant suffi à mettre en évidence que la grosse difficulté se situe plus en amont, lors du calcul des intersections entre les contours et de la nature de ces intersections. Il est en effet parfois difficile de savoir s'il y a effectivement intersection et, dans l'affirmative, s'il s'agit d'un contact ou d'une traversée même si l'on ne manipule, comme cela a été le cas pour nous, que des contours composés de segments de droites et d'arcs de cercles (fig 3.23). En fait, nous avons passé davantage de temps sur les modules d'intersection que sur l'algorithme d'union lui-même.

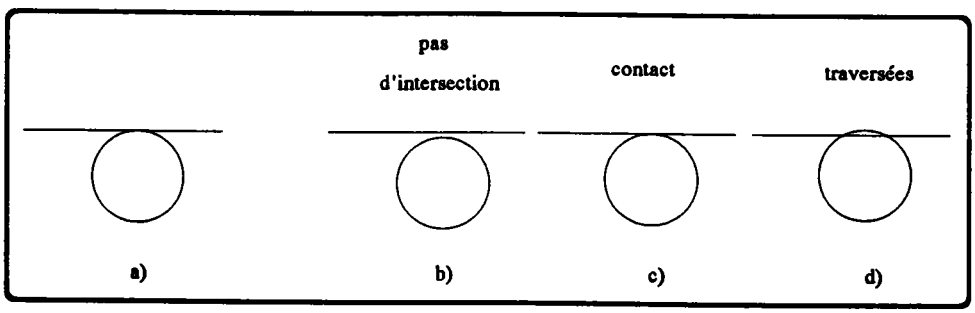


figure 3.23

La méthode peut être étendue à la composition de faces, éventuellement limitées par plusieurs contours. Les raisonnements employés jusqu'alors, ne s'appuyant que sur des

connaissances locales des intersections (nature, position de la matière, des brins incidents...), sont immédiatement applicables.

Pour être tout-à-fait complet, il faudrait se pencher sur d'autres aspects de la composition de contours:

- autoriser d'autres opérandes que des contours, en particulier des demi-espaces limités par une ligne ouverte.
- appliquer certains post-traitements aux contours calculés: en effet, il peut arriver d'y trouver des segments consécutifs colinéaires ou des arcs concentriques consécutifs de même rayon, ce qui peut géner des algorithmes s'appuyant sur ces contours (fig 3.24).

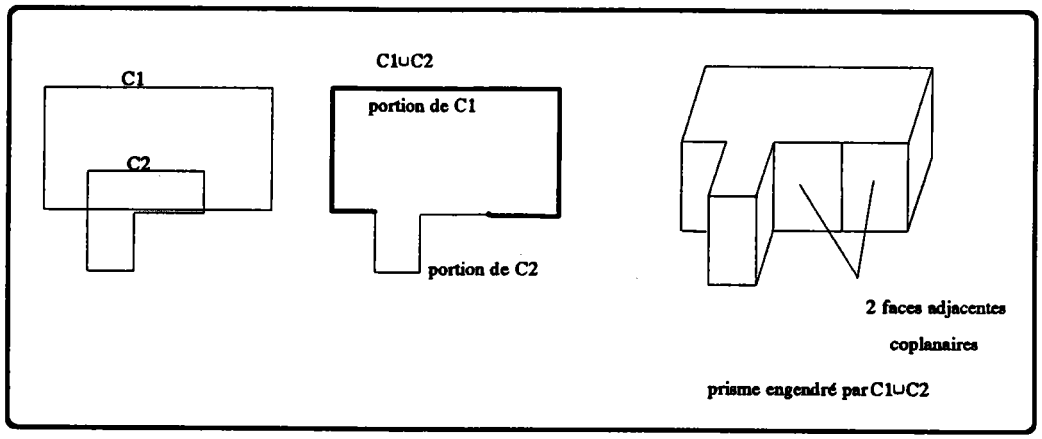


figure 3.24

3.3. Les opérateurs d'Euler

Un modeleur géométrique de solides s'appuyant sur une représentation B-rep est formé de plusieurs couches:

- un **noyau** constituant le modèle à proprement parler c'est-à-dire la maquette informatique de l'objet. Cette maquette, une **structure de données** du type faces, arêtes, sommets, comporte des informations géométriques (coordonnées des sommets, équations des faces) et topologiques (arêtes, faces...).
- les **primitives de base** d'accès au modèle: pour réaliser une indépendance des applications vis-à-vis de la structure de données (et non du type de représentation), il est indispensable d'interdire au développeur tout accès direct à la structure: celui-ci, pour initialiser, consulter ou modifier le modèle, utilise un jeu de primitives (ou fonction d'accès) fournies avec le modeleur. Ainsi, si la structure du modèle vient à changer, seules les primitives sont remises en question, les applications restent inchangées.

Tous ces opérateurs n'ont que de faibles possibilités de maintenir la validité de la représentation: il est clair, par exemple, que pendant toute la durée de la construction d'un objet, le modèle se trouve dans un état incohérent: la majorité des vérifications se fait à posteriori (comme le test d'appartenance d'une arête à exactement deux contours).

- les **primitives d'accès de haut niveau**, utilisant les primitives de base. Ce sont par exemple les opérations de création et de combinaison de volumes. C'est à ce niveau qu'est réalisée l'indépendance des couches supérieures vis-à-vis du type de représentation (B-rep, C.S.G....): si le choix de la représentation vient à changer, cette couche est (théoriquement) la dernière à subir des modifications.
- les **applications**, généralement écrites dans un langage s'appuyant entre autre sur des fonctionnalités de la couche immédiatement inférieure, mais aussi sur les primitives de haut niveau d'autres modèles (dont le modèle de dialogue).

Dans le domaine du maintien de l'intégrité du modèle, les formules d'Euler et d'Euler-Poincaré ont motivé des primitives de base assurant davantage de contrôle que les autres, plus classiques. Ces primitives, les opérateurs d'Euler [EAS 79] [MAN 82] [MAN 84] [WIL 85] [MAN 88], ont pour but de maintenir à tout instant le modèle dans un état topologique correct.

Les deux formules lient le nombre de sommets (S), le nombre d'arêtes (A), de contours intérieurs (CI), de faces (F), de composantes connexes (CC) et de trous (le nombre de trous est aussi appelé genre, d'où l'abréviation G); un trou est un perçage de part en part, non une simple dépression: il débouche donc de part et d'autre sur du vide. On a

(1) pour un objet non troué $S - A + F = 2CC$ (formule d'Euler)

(2) pour un objet troué $S - A + F - CI = 2(CC - G)$ (formule d'Euler-Poincaré)

La deuxième formule est une généralisation de la première; plutôt que de considérer deux cas, nous n'utiliserons donc que celle-ci.

Exemples:

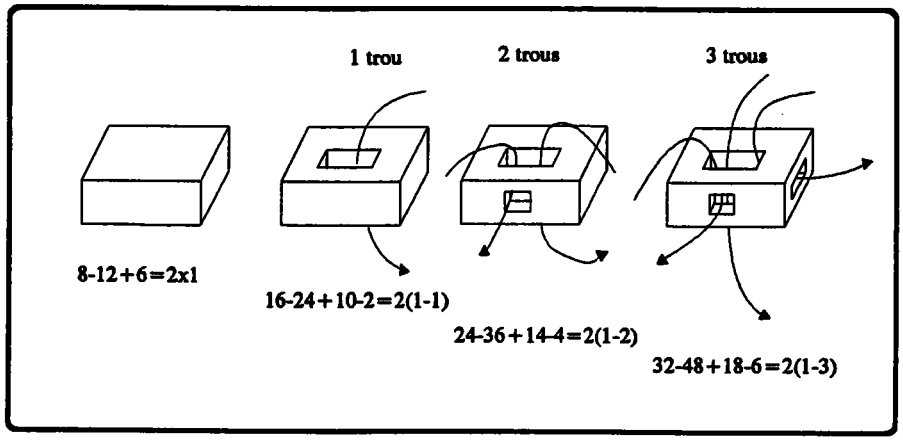


figure 3.25

Le principe des primitives de base évoquées ci-dessus est que si un objet vérifie la relation d'Euler-Poincaré avant leur intervention, il la vérifie toujours après. Ceci implique par exemple que si on lui ajoute une arête (A augmente de 1), il faut nécessairement qu'une ou plusieurs des autres grandeurs changent de manière à ce que l'équation reste équilibrée. En l'occurrence, on peut:

- incrémenter le nombre de sommets,
- incrémenter le nombre de faces,
- diminuer le nombre de contours intérieurs.

Ces trois opérations correspondent à trois opérateurs très courants utilisés dans la suite.

On assure ainsi qu'à n'importe quel stade de la conception, le modèle est valide *du point de vue des relations (1) et (2)*. En aucun cas, on ne peut attendre de quelconques vérifications géométriques: la figure 3.26 montre que ceci est insuffisant pour garantir la validité au sens où elle a été définie dans le chapitre un.

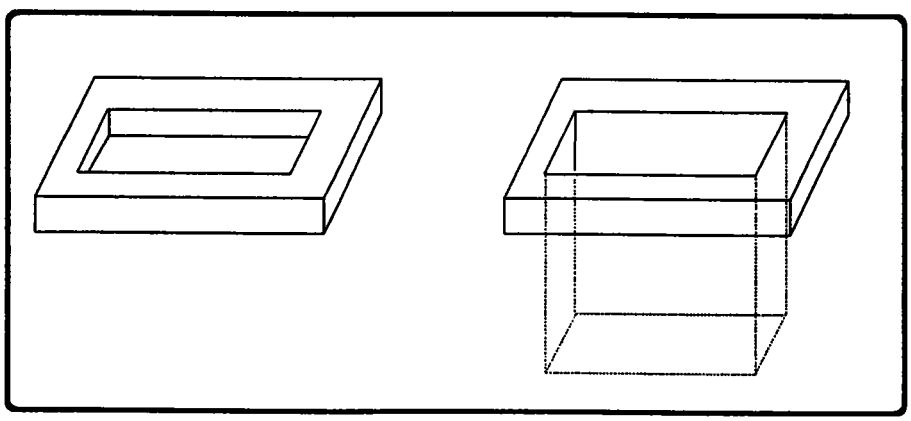


figure 3.26: deux objets de mêmes structures topologiques mais dont l'un est invalide

Dans la limite où les équations (1) et (2) restent équilibrées, on peut proposer un grand nombre de primitives d'accès au modèle (en création et destruction; les consultations restent à la charge d'opérateurs traditionnels). Pratiquement, on considère [MAN 88] que les cinq opérateurs cités ci-dessous (avec leurs inverses) suffisent. On montre en effet que tout objet satisfaisant à la relation d'Euler-Poincaré peut être construit à l'aide d'une séquence finie d'applications de ces cinq opérateurs [MAN 84].

Le nom des opérateurs est généralement soumis à une syntaxe proposée par Mäntilä. Nous respectons ici cette syntaxe car, bien que composée d'abréviations anglaises, elle est quasiment devenue un standard: chaque lettre du nom de l'opérateur correspond à un mot (verbe ou objet) et la suite des lettres décrit l'action de la fonction:

M : Make (créer)
K : Kill (détruire)

V: Vertex (sommet)
E: Edge (arête)
R: Ring (contour intérieur)
F: Face (face)
H: Hold (trou)
S: Solid (solide).

Les cinq opérateurs sont les suivants (consulter [PUC 87] et [MAN 88] pour les détails d'implantation en Pascal et C):

1) MVFS et KVFS

Partant de rien (équation 0+0+0+0 = 2.(0+0)), MVFS crée un solide satisfaisant l'équation (2), réduit à une face de un sommet (fig 3.27). Ceci ne correspond évidemment pas à un objet réel mais constitue un point de départ indispensable puisque la validité n'est garantie que si les règles d'Euler sont appliquées à des objets (et en particulier le premier dans l'historique de création) topologiquement corrects.

KVFS détruit un solide réduit à une face de un point.

figure 3.27

2) MEV et KEV

MEV crée un sommet de coordonnées données et une arête pour le relier à un sommet existant de numéro donné. Pratiquement, les choses se passent en deux temps:

- on détache du sommet existant certaines des arêtes incidentes (la sélection est faite d'après des paramètres supplémentaires de l'opérateur) (fig 3.28.a).
- on attache une nouvelle arête au sommet existant, le nouveau sommet à l'extrémité de la nouvelle arête et les arêtes détachées au nouveau sommet (fig 3.28.b).

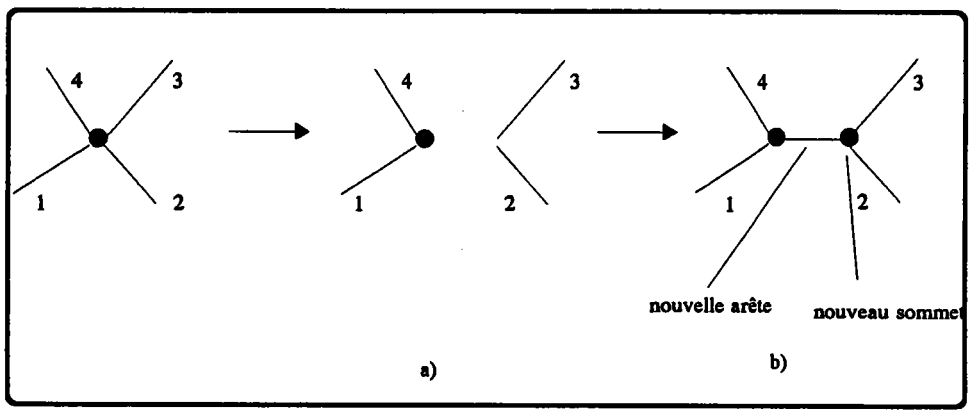


figure 3.28

Deux cas particuliers constituent l'essentiel de l'utilisation de cette primitive:

- le sommet n'a aucune arête, ce qui est le cas lorsque l'on crée le volume initial réduit à un point.
- aucune arête n'est détachée du sommet existant, on se contente simplement de lui en attacher une nouvelle.

KEV détruit une arête donnée en réunissant ses deux sommets en un seul et en faisant converger sur cet unique sommet les arêtes qui arrivaient sur les deux anciens sommets.

3) MEF et KEF

MEV insère une arête entre deux sommets d'une face et scinde ainsi la face en deux autres faces (fig 3.29).

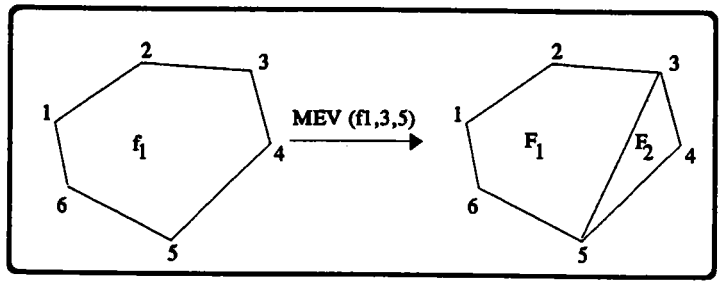


figure 3.29

KEF réunit deux faces en une seule en détruisant leur arête commune.

4) KEMR et MEKR

KEMR détruit dans le contour d'une face une arête donnée qui doit y apparaître deux fois. Ceci a pour effet d'isoler une boucle qui constitue un contour intérieur (fig 3.30).

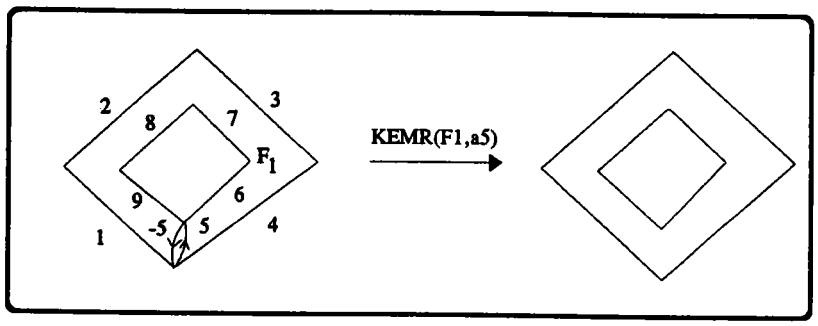


figure 3.30

MEKR insère une arête allant d'un sommet du contour extérieur vers un sommet d'un contour intérieur, réunissant ainsi les deux boucles en une seule.

5) KFMR et MFKR

KFMR détruit une face F1 en tant que telle et la transforme en un contour intérieur d'une autre face F2.

Cet opérateur permet de réaliser:

- le collage de deux solides si F1 et F2 appartiennent à des objets différents (fig 3.31.a).
- la création dans un solide d'un trou de part en part ou d'une anse, si F1 et F2 sont des faces d'un même objet (fig 3.31.b).

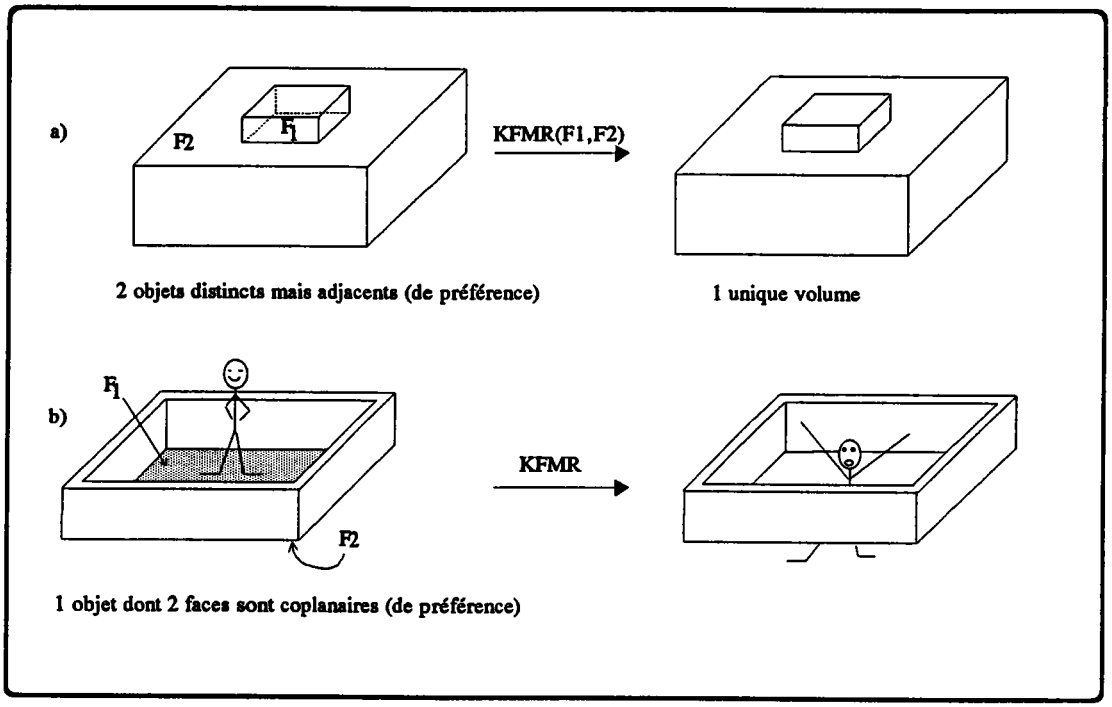


figure 3.31

Dans les deux cas, il est préférable, pour la vraisemblance de la construction, que les deux faces soient coplanaires ou au moins, si le B-rep supporte les faces gauches, sur la même surface. Ceci n'a cependant aucune influence sur le fonctionnement de l'opérateur qui ne se préoccupe que des aspects topologiques.

MFKR est l'opérateur inverse de KFMR.

A titre d'illustration, nous détaillons dans le paragraphe suivant les étapes de la construction d'un objet de révolution.

3.4. Mise en oeuvre de l'opérateur de révolution

Ce paragraphe est consacré à la mise en oeuvre de l'opérateur de révolution dans deux cas particulier:

1) le générateur est une ligne plane ouverte touchant l'axe en ses deux extrémités et nulle part ailleurs.
2) le générateur est une ligne fermée située dans un même plan que l'axe mais sans contact avec ce dernier.

L'objectif est ici de construire un solide bordé par la surface de révolution et, lorsque l'angle de rotation est inférieur à 360°, par le générateur à ses positions initiale et finale (fig 3.32). Des exemples de ces constructions (copies d'écran) sont donnés dans la figure 3.32 et sur la planche 2 (en fin du paragraphe 3.4).

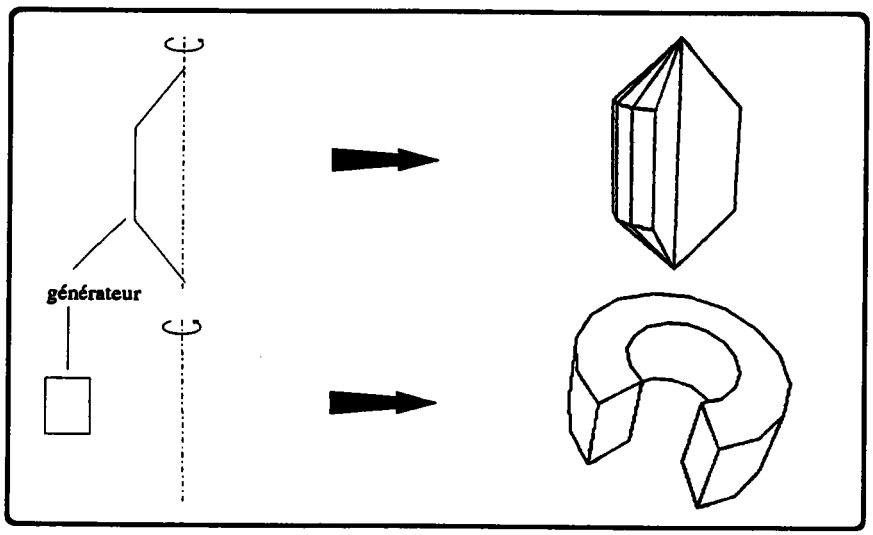


figure 3.32

Dans les deux cas, les accès au modèle géométrique sont réalisés au travers des opérateurs d'Euler. Ceci permettra une discussion de leurs avantages et inconvénients en 3.5.

Nous avons dit que le modèle utilisé est un B-rep polyédrique: c'est donc une approximation du générateur que l'on passera aux deux algorithmes ci-dessous, soit une ligne brisée, fermée dans le deuxième cas.

Nous ne détaillons pas les aspects géométriques de la rotation: entre autres, nous supposons connues les coordonnées des sommets de l'approximation afin de nous concentrer sur l'aspect topologique de la création [MIN 88].

3.4.1. Révolution d'une ligne ouverte

Pour simplifier les explications, nous supposerons ici que les premier et dernier segments du générateur sont perpendiculaires à l'axe: on diminue ainsi le nombre de cas à exposer sans pour autant réduire la difficulté (fig 3.33.a).

La nuance vient de ce que dans l'hypothèse d'un segment oblique, le premier segment engendre une surface conique qu'il faut approcher par une série de faces triangulaires alors qu'avec nos conventions, il n'engendre qu'une face circulaire plane.

Nous distinguerons toutefois trois cas: celui d'un angle de rotation plein (360°), plat (180°) ou quelconque (fig 3.33 b,c et d). Les figures à suivre illustreront donc toutes trois cas.

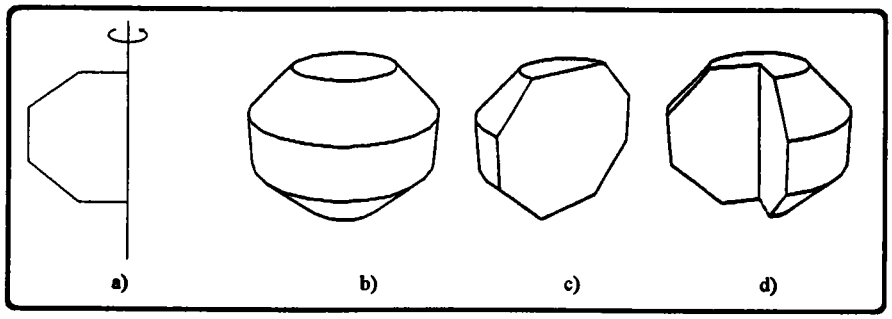


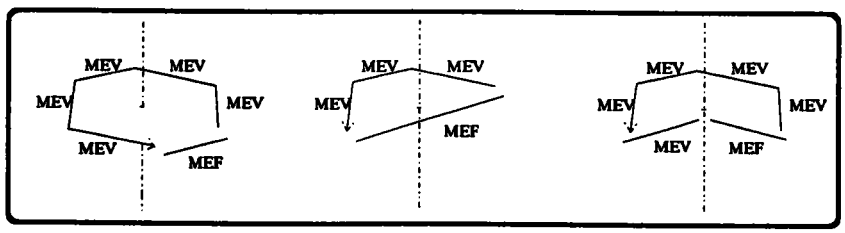
figure 3.33

Les étapes de la construction sont les suivantes:

- **prétraitements**: calcul du nombre nbPts de points sur un parallèle.

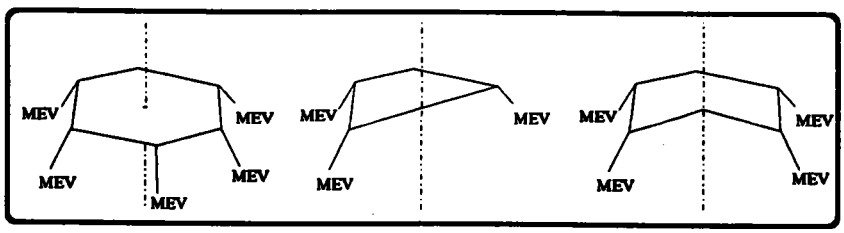
- **création de la face du dessus**:
  - . création du parallèle par une série de (NbPts) MEV dans l'unique face du volume.
  - . fermeture de la face avec, selon les cas, un MEF (angle de 360° et 180°) ou un MEV (pour aller jusqu'au centre) et un MEF (angle quelconque).

    On obtient alors un solide à deux faces, superposées comme le sont le recto et le verso d'une page.

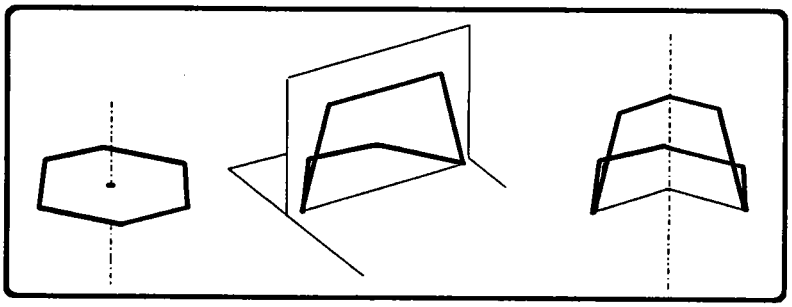


- **création du principal de la surface**:

  Dans la deuxième face, on crée, par NbPts appels à MEV, les arêtes qui feront la liaison avec le prochain parallèle.



Par une série de MEF, on relie deux à deux les extrémités de ces arêtes, créant ainsi une bande de facettes. La face du fond a alors cet aspect:

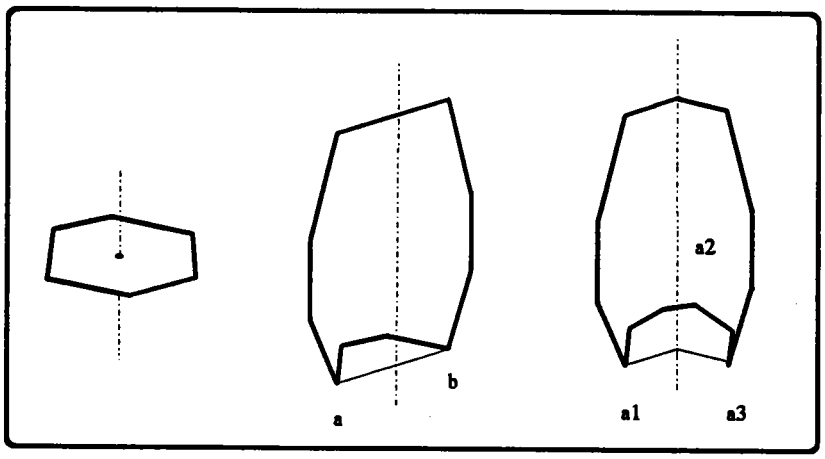


Dans les deux derniers cas, et bien que cela puisse choquer, elle a donc un contour non plan (c'était d'ailleurs également le cas au pas précédent).

Ces deux points sont répétés pour tous les sommets du générateur, jusqu'à l'avant dernier.

- **post-traitements:**

A ce stade, la face du fond a l'aspect suivant:



Dans le premier cas, elle est donc parfaite telle qu'elle est et la construction est terminée.

Dans le deuxième, il suffit de la scinder en créant, par un appel à MEF, une arête entre les sommets a et b. La construction est alors terminée.

Dans le dernier, il faut d'abord créer une arête simple a_1 avec MEV puis, à l'aide d'une nouvelle arête a_2, couper la face en deux avec MEF et enfin, avec un dernier MEF (a_3), recouper l'une des "moitiés". On a alors les trois faces qui ferment le volume.

- **modélisation de la surface torique**: par une suite de MEV, on crée des isthmes dans F2; en reliant les extrémités de ces isthmes (MEF), on crée une première bande de facettes (fig 3.39).

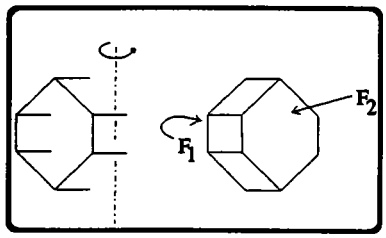


figure 3.39

Le processus est répété jusqu'à ce que la rotation soit terminée. A ce stade, si l'angle de rotation est de moins de 360°, la création est terminée, toutes les facettes sont créées.

- si par contre l'angle de rotation est plein, la dernière face Fn est confondue géométriquement avec F1: **cette superposition doit apparaître dans la topologie** et impose un collage de F1 à Fn.

Dans un premier temps, nous exposons donc une opération de collage entre deux faces exactement superposées, en utilisant une méthode que nous avions implantée et dont le principe a été corroboré par Mäntilä dans [MAN 88]. Nous proposons ensuite une méthode plus simple qui s'applique tout-à-fait au cas de révolution qui nous intéresse.

Détaillons donc le collage de Fn à F1:

. il sera nécessaire de relier par des arêtes (de longueur nulle) les paires de sommets confondus; or l'opération de création d'arête utilisée à cette fin (MEF) ne peut relier des sommets que d'un même contour. Il faut donc réunir dans une même face les sommets de F1 et de Fn. Ceci est réalisé avec KFMR qui fait de Fn un contour intérieur de F1 (pour clarifier l'illustration, Fn est représentée dans la figure 3.40 d'une taille légèrement inférieure à celle de F1) puis avec MEKR qui réunit les deux contours en un seul par adjonction d'une arête de liaison (fig 3.40).

De prime abord, cette façon d'aborder les choses, en particulier le passage par des faces non planes, peut choquer.

Pourtant, on peut trouver une certaine élégance à ces opérateurs: par la méthodologie qu'ils imposent, à savoir de créer une première face, de la scinder, de déformer l'une des moitiés etc, ils évoquent un travail créatif et évolutif qui peut séduire.

Pour rester tout-à-fait prosaïque, il faut aussi noter qu'ils empêchent de faire n'importe quoi n'importe quand: il est très malaisé, par exemple, de créer dans un premier temps tous les sommets du solide puis ses arêtes et enfin ses faces: ils contraignent donc à une certaine démarche et, par là, à une certaine rigueur.

3.4.2. Révolution d'un contour

La rotation d'un contour autour d'un axe engendre une surface torique. Si l'angle de rotation est de 360°, elle est automatiquement fermée et suffit à border le volume que nous voulons construire. Dans le cas contraire, il faut lui adjoindre deux faces planes limitées par les première et dernière instances du générateur.

De ces remarques, on tire les étapes de la construction:

- **modélisation de la première face de fermeture**, limitée par le générateur à sa position initiale. Cette étape, nécessaire même si la rotation est de 360°, est réalisée par un appel à MVFS, des appels successifs à MEV et un appel à MEF (fig 3.38).
  A ce stade, on a deux faces superposées (comme le sont le recto et le verso d'une feuille) dont l'une va être déformée en surface torique.
  Remarquons que les deux faces en question ne sont superposées (et incorrectes d'un point de vue géométrique) que parce que l'on suppose que les facettes sont planes: dans une approche plus générale, elles pourraient être des calottes sphériques et limiter un volume géométriquement valide.

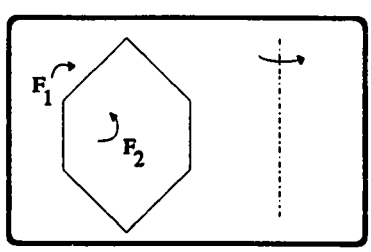


figure 3.38

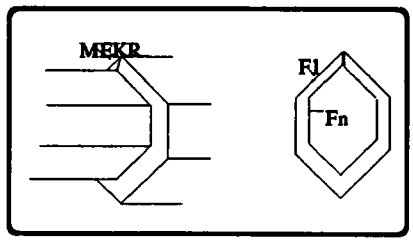


figure 3.40

. on peut alors relier les autres paires de points confondus par autant d'arêtes (MEF), créant ainsi un certain nombre de faces d'aire nulle (fig 3.41).

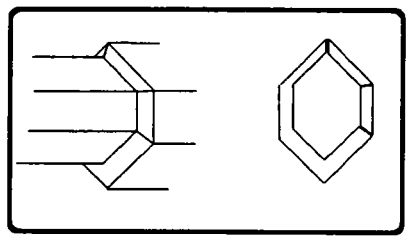


figure 3.41

. en détruisant (KEF) les arêtes de ce qui était Fn, on réunit les faces de la dernière bande de facettes aux faces nouvellement créées (fig 3.42).

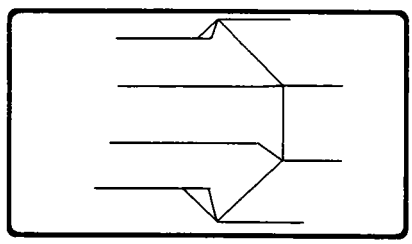


figure 3.42

. il ne reste qu'à réunir topologiquement les sommets de Fn à ceux de F1 en détruisant par des appels à KEV les arêtes qui les relient (fig 3.43).

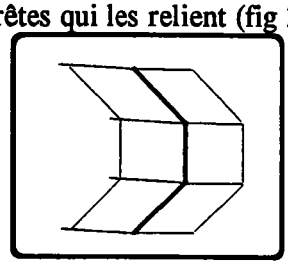


figure 3.43

Nous voyons à présent que la fermeture du tore peut être réalisée beaucoup plus facilement si l'on arrête la création des méridiens à l'avant dernier. Celui-ci n'est pas

superposé au premier méridien et les choses s'en trouvent grandement facilitées. En reprenant les notations précédentes, considérons deux faces Fn et F1 (fig 3.44).

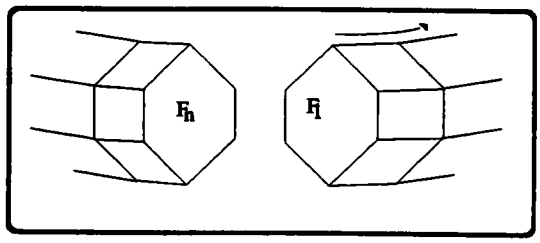


figure 3.44

Bien qu'elles ne soient pas coplanaires, on peut faire de Fn un contour intérieur de F1 (KFMR): F1 est alors une face bordée par deux contours non coplanaires, que l'on réunit dans une même boucle à l'aide de KRMH (fig 3.45).

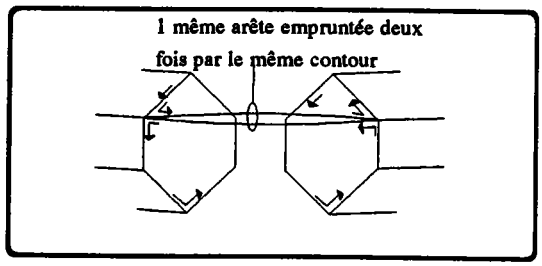


figure 3.45

Il suffit maintenant, à l'aide d'une série de MEF, de lier par des arêtes les sommets en vis-à-vis sur ce qui était Fn et ce qui était F1. F1 est ainsi compartimentée en plusieurs faces qui constituent la dernière bande de facettes du tore (fig 3.46).

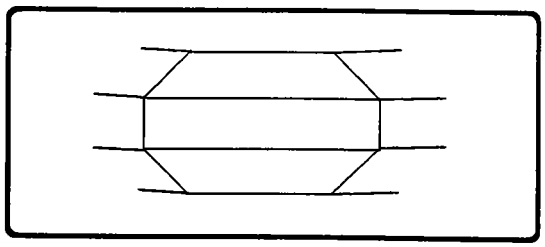
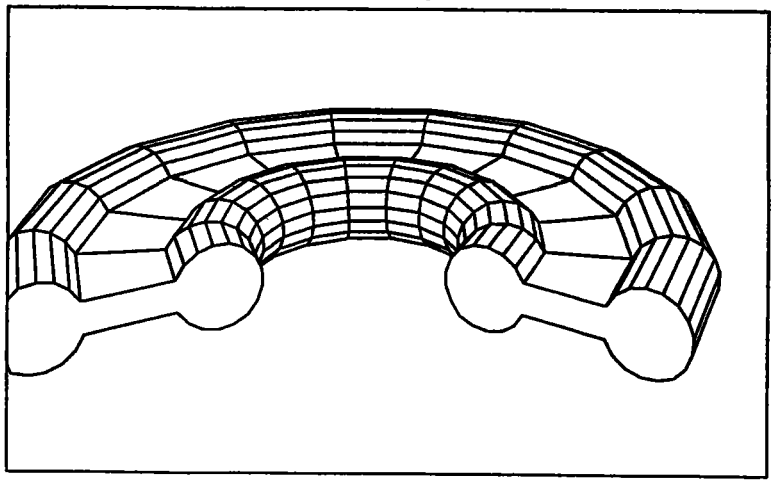
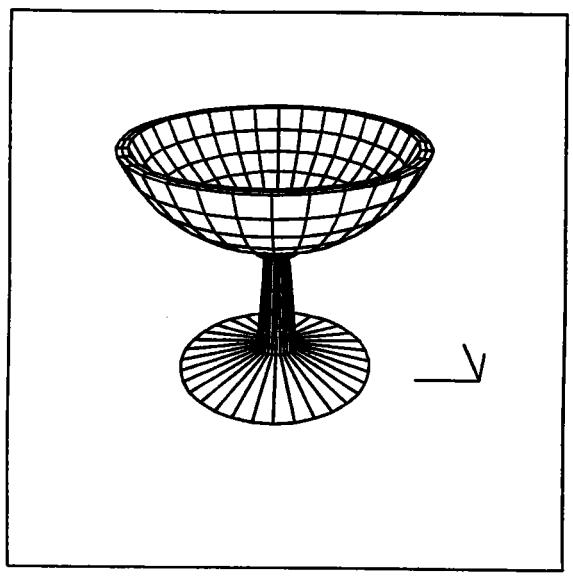


figure 3.46





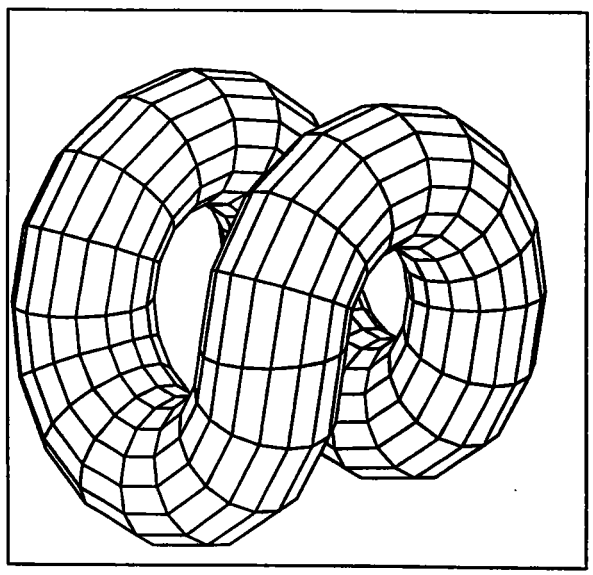


PLANCHE 2: les deux premiers exemples correspondent à la révolution d'une ligne ouverte ou fermée.
Le troisième est construit par juxtaposition de demi-tores.

3.5. Critique des opérateurs d'Euler

A la lumière des deux paragraphes précédents, résumons quelques qualités et défauts des opérateurs d'Euler dans un court bilan.

Nous leur avons trouvé deux qualités importantes:

- ils amènent une certaine élégance dans la programmation et contraignent à une certaine rigueur.
- à tout stade de la construction, ils garantissent la validité topologique (au sens de la relation d'Euler-Poincaré) de la pièce.

En contrepartie, ils limitent les possibilités du développeur en ce sens qu'il ne peut plus manipuler le modèle aussi librement qu'avec des primitives d'accès traditionnelles. Ce peut être ressenti comme une perte - par contre, la puissance du B-rep n'est pas remise en cause: on a dit que tout polyèdre est obtenu par une séquence finie d'appels aux opérateurs d'Euler -

L'élégance est affaire de goût: on aime ou on n'aime pas.

La rigueur est, elle, une nécessité mais on ne peut pas dire qu'elle soit l'apanage des opérateurs d'Euler: on peut très bien adopter des conventions strictes sur les modes d'accès traditionnels au modèle. Nous avons d'ailleurs été encouragés dans cette deuxième voie en comparant les temps d'accès au modèle avec les deux approches: nous avons constaté une durée au moins trois fois supérieure avec les opérateurs d'Euler (ces mesures s'appuient sur une programmation des opérateurs d'Euler que nous pensons correcte mais qui est peut-être perfectible). Or, pour les algorithmes de révolution cités précédemment, on répartit ainsi la durée de la création:

- entre 10 et 20 % pour les calculs (position des points constituant l'approximation polyédrique.
- le reste pour l'initialisation du modèle.

La majorité du temps est donc passée en accès au modèle et la relative lenteur des opérateurs d'Euler en ce domaine constitue un handicap.

L'argument fort en leur faveur est la garantie d'une topologie correcte à tout stade de la création. A priori, et pour deux raisons, ceci ne devrait pas leur être un plus:

- lorsqu'un algorithme est intégré dans un système, il est censé avoir été suffisamment pensé pour garantir la validité de ses produits: une précaution supplémentaire devrait donc être inutile.

- il importe peu que dans les étapes intermédiaires la validité topologique soit assurée puisqu'à ce stade, l'objet est de toutes les façons inutilisable pour des applications pratiques (affichage, simulations...). Dans certains cas, par exemple les opérations booléennes, il peut en outre être extrêmement contraignant d'imposer que lors de toutes les étapes intermédiaires, le modèle soit valide du point de vue topologique.

Dans le cas d'algorithmes dont on maîtrise parfaitement tous les aspects, le premier point semble effectivement être justifié. On n'évolue cependant pas toujours dans un contexte simple et il peut alors être précieux d'appliquer un contrôle supplémentaire.

Pour atténuer l'effet négatif du deuxième point, on peut avancer que si tel est le prix d'une garantie de la topologie en fin de création, alors ce n'est pas cher payé: il y a ici un choix à faire dépendant essentiellement de la difficulté de l'algorithme à mettre en place et de la manière dont on pense le dominer.

Finalement, tout particulièrement dans le cas des algorithmes de révolution dont on cerne parfaitement tous les aspects, il semble que l'apport des opérateurs d'Euler ne justifie ni le changement qu'ils imposent dans les méthodes de travail, ni les complications qui apparaissent dans des cas particuliers.

3.6. Intégration dans un système constructif paramétré

Les travaux de développement évoqués précédemment s'inscrivent dans le cadre du projet SACADO (Système Adaptatif de Conception Assistée et de Développement par Ordinateur) dont la vocation est avant tout de poser les bases d'un système interactif et adaptatif [GAR 88]. En tant que tels, ils ont fait l'objet de quelques réflexions visant à rendre plus confortable leur utilisation, en particulier dans le domaine du paramétrage.

Pratiquement, un objet extrudé est la donnée:
- d'un contour
- d'informations 3D.

Son paramétrage est donc réalisable à deux niveaux.

Les informations 3D sont un vecteur (prisme), un axe (révolution) ou une trajectoire (cylindre généralisé). Leurs grandeurs caractéristiques (points de passage, longueur, direction) sont généralement fournies par un opérateur humain ou calculées par contrainte (point à l'intersection de deux objets, segment parallèle à un autre segment, courbe passant

par n points, eux-mêmes calculés sous contrainte...). Leur paramétrage, très proche de la notion d'historique, passe donc par une facile mémorisation de la nécessité de données interactives et par une très difficile modélisation des contraintes [JUN 90].

La deuxième possibilité de paramétrage vient du contour lui-même dont on ne peut pourtant pas vraiment dire qu'il est paramétrable: il est plus juste de dire que les objets sur lesquels il s'appuie le sont eux-mêmes (de la même manière que les vecteurs, axes et trajectoires ci-dessus) et que ceci est répercuté par des notions d'historique sur le contour proprement dit.

Pour mesurer, entre autres, l'ampleur des modifications induites par un changement sur un objet support, un système de cotation convivial et à sémantique forte est indispensable. Outre les opérateurs d'extrusion, nous avons donc mis en oeuvre un système de cotation obéissant aux principes suivants:

- absence de menus: les objets de base traités dans SACADO 2D sont les points, les segments, les cercles et arcs de cercle; pour être complète, une arborescence de menus permettant de coter un ou deux objets devrait donc laisser le choix entre 14 options principales (point, point-segment, point-cercle,...) et d'autres, complémentaires (distance, DX ou DY pour une cotation entre deux points, par exemple).
  Pour soulager l'opérateur, nous avons préféré une approche que nous pensons plus attrayante et qui ne demande à l'opérateur, après avoir sélectionné le menu global de cotation, que de montrer le premier objet à coter puis, s'il existe, le deuxième.
  Le système détermine automatiquement, à partir de la nature des objets désignés, le cas de cotation correspondant. Ceci est rendu possible par la qualité du moniteur de dialogue utilisé dans SACADO [GAR 88] [TOT 89].
- les tracés et la valeur de cotation sont alors affichés, mais restent soumis au déplacement de la souris. Ceci permet, notamment pour des raisons esthétiques, de déplacer interactivement (en affichage élastique) l'ensemble de la cotation jusqu'à ce qu'elle soit considérée comme bien placée.
- à tout moment, il est possible de faire défiler les différentes cotations possibles sur le ou les deux objets désignés, par simple pression sur la barre d'espacement: le nombre de ces sous-cas étant toujours inférieur à 6, cela constitue une solution très satisfaisante.
- à tout moment, il est également possible de taper d'autres caractères, qui remplacent le pré et le post textes attachés à la valeur de cotation.
- ces opérations sont les actions possibles lors de la création ou de la modification interactive d'une cotation. L'opérateur peut en outre forcer la valeur d'une cotation à

une valeur de son choix. Un intérêt de cette option est qu'il peut ainsi faire un plan d'une manière rapide et relativement imprécise ("à main levée") puis coter les éléments importants et forcer certaines valeurs de manière à ce que les utilisateurs futurs du plan ne se basent pas sur des grandeurs erronées du fait que le dessin a été fait sans soucis de précision.

- les cotations sont associatives: le déplacement ou la modification d'un objet coté provoquent une réévaluation de la cotation, qui tient également compte d'un changement automatique de cas de cotation.

Exemple: Supposons qu'existent deux segments sécants cotés, la valeur de la cote est la valeur du secteur formé par les deux droites.
Si l'on fait subir à l'un des segments une transformation telle qu'il devienne parallèle à son vis-à-vis,
alors la cotation devient automatiquement la distance entre ces deux segments; la valeur et les tracés changent en conséquence.

Le système de cotation tel que présenté ci-dessus est effectivement intégré au système SACADO. On trouvera dans l'annexe 3 [GAR 88] un article paru dans MICAD 88 brossant les grands traits du dialogue dans ce logiciel et montrant d'une part la convivialité et d'autre part que ce paramétrage peut être facilement introduit. Le dialogue et l'implantation sont basés sur des outils que nous ne détaillons pas dans ce chapitre, mais dont l'annexe montre les fondements.

3.7. Conclusion au troisième chapitre

Cette troisième partie vise à donner une idée de notre participation pratique au projet SACADO. Elle détaille donc certaines des fonctions que nous y avons intégrées.

La première est une fonction de création d'une classe de volumes que nous appelons prismes généralisés. Nous ne revenons pas sur les améliorations à apporter à la version actuelle, exposées en 3.1.3, ni sur les nécessaires extensions: nous en sommes conscients mais pensons aussi que d'autres thèmes doivent être abordés, auxquels nous jugeons préférable d'attribuer une plus grande priorité: cette question fait l'objet de la conclusion générale du mémoire.

Il est bien sûr difficile d'affirmer ou de prouver que l'algorithme existant est en tous points correct; en effet:

- il est clair, tout d'abord, qu'il est extrêmement ardu de prouver un programme de cette nature, en particulier à cause des aspects géométriques.
- les tests qui ont été effectués l'ont été à petite échelle, ceci pour deux raisons:
  - ils n'ont été pratiqués que par un petit nombre de personnes (quelques utilisateurs occasionnels en plus de l'auteur).
  - le développement sur micro-ordinateur a fait que le volume de mémoire dont nous disposions était faible et que, par conséquent, les modèles conçus n'ont jamais été de grande taille.

  Malgré cela, nous pensons tout-de-même que les essais auxquels nous avons procédés sont significatifs et qu'ils confirment le résultat de nos réflexions.

  D'autre part, ces deux limitations disparaîtront bientôt puisqu'un transport vers des machines de plus grande capacité (Apollo et Sun) est en cours (ce qui lèvera les problèmes de limitation de mémoire) et que nous comptons faire travailler des étudiants sur ces versions, comme c'est actuellement le cas pour SACADO 2D.

La deuxième fonction décrite est l'opérateur de combinaison de contours, qui est une étape nécessaire lors du collage de deux volumes. Nous avons vu que cette étude amenait à distinguer un certain nombre de cas (une trentaine). Nous pensons qu'il est une bonne chose de ne pas compacter cette liste car on peut ainsi modifier à loisir le comportement de l'algorithme sans intervention dans la programmation.

Nous nous sommes ensuite attardés sur des opérateurs d'accès au modèle B-rep garantissant la validité topologique des volumes construits. Ces fonctions d'accès, les opérateurs d'Euler, ont été utilisées pour la modélisation d'objets de révolution. Le détail des étapes de la construction a permis une discussion de ces opérateurs et nous sommes finalement arrivés à la conclusion qu'un recours à ces primitives ne se justifiait pas dans le cadre de la mise en oeuvre d'algorithmes relativement simples.

Nous avons terminé en évoquant la fonction de cotation 2D et en essayant de montrer dans quelle mesure le lien existant entre les objets 3D et les contours 2D sur lesquels ils s'appuient se rapproche de la notion de paramétrage.

Nous avons volontairement omis d'évoquer le matériel de base impliqué dans ces fonctions, matériel qu'il a fallu développer mais que nous n'avons pas jugé utile de décrire.

CHAPITRE IV

4. EXTRUSION ET EXTRACTION DE CARACTERISTIQUES DE FORME

4.1. Objectif

Parmi les nombreux problèmes encore ouverts de la C.F.A.O., l'intégration conception/fabrication est une étape indispensable dans le développement de systèmes industriels.

Dans ce domaine, un axe de recherche consiste à mettre en évidence, dans le modèle informatique produit par la conception, certaines informations technologiques utiles à la fabrication, effectivement présentes dans le modèle mais *d'une manière implicite seulement.* En considérant les modèles géométriques, les informations en question sont ce que l'on appelle des *caractéristiques de forme (form features)* c'est-à-dire des zones de la pièce présentant un intérêt pour l'un quelconque des secteurs de la production. Dans ce qui suit, nous nous intéresserons plus spécialement aux caractéristiques touchant au secteur fabrication (fig 4.1).

Grâce à elles, on pense en effet arriver à diminuer les interventions humaines nécessaires à l'interprétation des plans ou des modèles fournis par la conception et à la rédaction des programmes de fabrication [GAM 90].

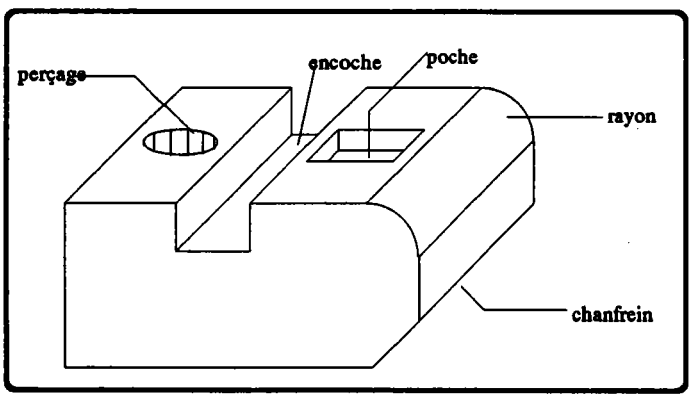


figure 4.1: quelques caractéristiques de fabrication auxquelles on associe un enlèvement de matière (fraisage, perçage)

A certaines de ces caractéristiques, on peut associer un processus de fabrication par enlèvement de matière. Or qui dit enlèvement de matière dit trajectoire d'outil et volume balayé par un outil, donc extrusion. Il apparaît donc qu'existe un lien entre l'extraction de caractéristiques de forme et l'extrusion, lien que nous essayons de mettre en évidence dans les lignes qui suivent.

Pour résumer, disons que l'extrusion permet traditionnellement de créer des objets dans le modèle alors que, dans cette approche, on part du modèle pour en déduire des extrusions, à savoir des couples (*outil,trajectoire de l'outil*).

Nous n'avons comme objectif ni de résoudre le problème de l'intégration conception-fabrication, ni de déterminer les extrusions qui permettent d'usiner la pièce modélisée. En particulier, le calcul de la trajectoire de l'outil est une opération compliquée qui sort du cadre de la réflexion que nous nous sommes fixé.

Nous visons simplement à mettre en évidence, parmi les méthodes d'extraction, celles qui pourraient se prêter à ce genre de traitements.

Pour soulager l'écriture, nous utiliserons indifféremment les termes *caractéristique de forme* et *feature*, bien qu'il s'agisse d'un raccourci un peu abusif.

4.2. Méthodes d'extraction de features - portée des méthodes

Nous passons en revue, dans cette partie, les principales méthodes de détection de caractéristiques de forme dans un modèle (du type B-rep ou C.S.G.) et d'intégration, dans un modèle étendu, des informations calculées.

Les méthodes de détection varient avec le type de représentation:

- avec un modèle par les limites, on utilisera surtout les relations d'adjacence entre faces (informations topologiques...) et les angles entre ces dernières (... et géométriques). En parlant d'une arête droite, on dira qu'elle est convexe (resp. concave) si la matière y fait un angle de moins (resp. plus) de 180° (fig 4.2).

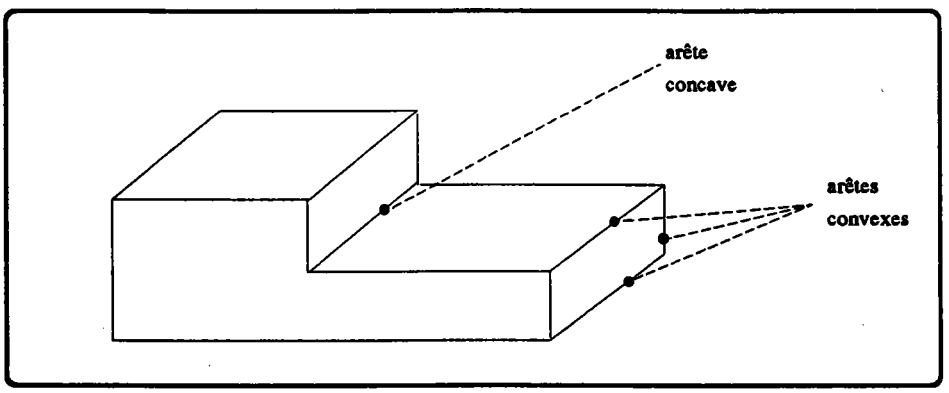


figure 4.2: arêtes convexes et concaves

- dans un représentation C.S.G., on cherchera à restructurer l'arbre de manière à faire apparaître des opérations de différence, souvent assimilables à des enlèvements de matière.

4.2.1. Méthodes associées aux modèles par les limites.

Les études les plus nombreuses se rapportent aux modèles par les limites et, pour être précis, aux modèles polyédriques.

Dans tous les cas, on se sert à des degrés divers des notions topologiques désormais courantes dans ces représentations:

- liaisons face - contours (extérieur et intérieurs),
- liaisons contour - arêtes,
- liaisons face - face (partage d'une arête ou d'un sommet),

et de certaines informations géométriques:

- convexité -concavité- d'une arête (la matière y fait un angle de moins -de plus - de 180 degrés),
- intersections d'une droite (support d'une arête) avec une autre droite ou un plan.

Les méthodes s'appuyant exclusivement sur les relations face - contours [FLO 89] donnent de bons résultats pour la détection de certaines cavités et protubérances, des anses et des ponts.

Celles qui n'exploitent que la concavité des arêtes [JOS 88] sont capables d'isoler les marches et les encoches mais, dans une moindre mesure, les trous et les anses.

Certaines, enfin, tachent de cumuler les deux avantages en exploitant tour à tour la topologie et la géométrie [FAL 87].

Nous détaillons dans la suite les principales méthodes.

4.2.1.1. Utilisation des relations face - contours [FLO 89]

Dans cette approche, le premier objectif est de décomposer la liste des facettes de l'objet (fig 4.3.a) en plusieurs sous-listes. Cette séparation se fait le long des contours intérieurs dans une opération qui est en quelque sorte l'inverse d'un collage et qui détermine un certain nombre de composantes dont quelques-unes sont des volumes "ouverts" (fig 4.3.b).

Ces derniers sont fermés par adjonction d'une face bordée par le contour le long duquel la séparation s'est faite (fig 4.3.c).

Dans un modèle B-rep, ces deux manipulations sont réalisées très simplement, par exemple avec l'opérateur d'Euler KRMF (Kill Ring-Make Face).

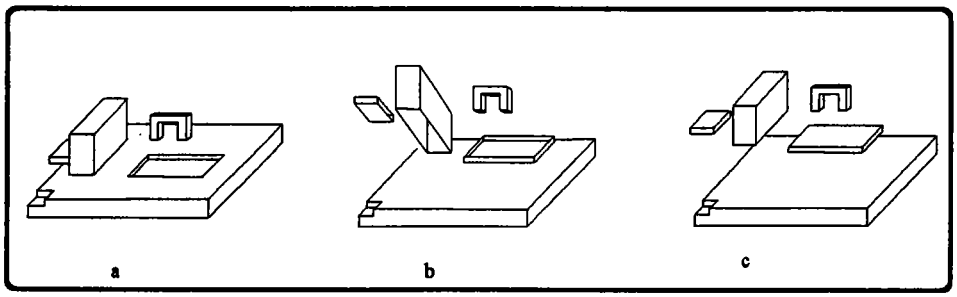


figure 4.3: la décomposition d'un polyèdre; noter que la marche dans le coin de la boite de base n'intervient pas dans la décomposition: ce n'est pas un feature.

Les volumes dont aucune face n'est limitée par un ancien contour intérieur sont les composantes principales de l'objet, celles qui lui donnent son apparence générale; les autres définissent les aspects locaux de sa forme.

Pour reconnaître parmi celles-ci les features, on établit un graphe orienté selon les règles suivantes:

- les noeuds sont les composantes volumiques isolées à l'étape précédente.
- un arc relie une composante C1 à une composante C2 (dans ce sens) si un contour de C1 incluait dans l'objet de départ, un contour de C2 (fig 4.4); il doit y avoir autant d'arcs que d'inclusions, donc que de contours intérieurs sur la pièce initiale.
  Un arc témoigne ainsi d'une adjacence entre deux volumes issus de la décomposition et est orienté du "portant" vers le "porté".

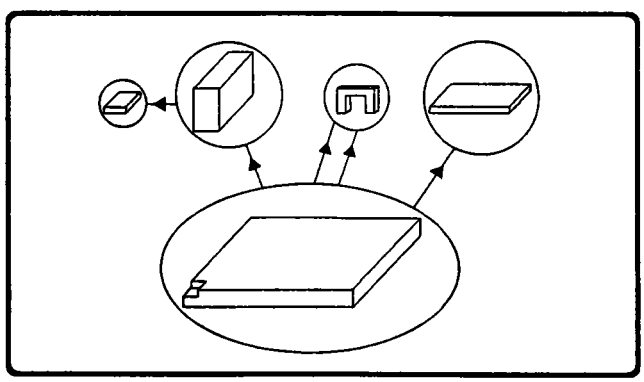


figure 4.4: graphe de l'objet de la figure 4.3

L'identification des features se fait alors de manière simple à l'aide de règles du type:

- un unique arc de C_1 vers C_2 implique que C_2 est une dépression ou une protubérance dans/sur C_1
- deux arcs ou plus de C_1 vers C_2 font de C_2 une anse ou un trou de C_1.

...

Notons qu'aucune information géométrique n'a été utilisée. Ce n'est qu'à ce stade qu'elles interviennent pour distinguer les sous-cas, et encore suffit-il de tester la concavité d'une unique arête pour séparer la protubérance de la dépression, l'anse du trou etc...

La méthode a en outre l'avantage d'être robuste et de traiter aussi bien des caractéristiques isolées qu'imbriquées (voir dans la figure 4.3 une protubérance sur une protubérance).

Si l'objet est limité par des faces maximales, le graphe fournit enfin une décomposition unique en composantes volumiques dont certaines sont des features. Ceci facilite la comparaison d'objets et permet une classification des pièces selon des critères technologiques.

La limitation de la méthode est dans sa portée, restreinte aux features introduits par des contours intérieurs: on manque ainsi des caractéristiques auxquelles il est possible d'associer des processus de fabrication (encoches, marches).

4.2.1.2. Utilisation de la concavité-convexité des arêtes [JOS 88]

Cette deuxième approche consiste elle aussi à établir un graphe dont:

- les noeuds sont les faces de l'objet.
- les arcs sont *valués* et *non orientés*: un arc témoigne du partage d'une arête par les deux faces qu'il relie et vaut 1 ou 0 suivant que cette arête est convexe ou concave.

En convenant du fait qu'une face ne comportant que des arêtes convexes ne peut intervenir dans un feature, le graphe est simplifié par suppression des noeuds dont tous les arcs incidents valent 1.

Ceci a pour effet d'isoler un certain nombre de composantes connexes (fig 4.5) que l'on compare à des graphes génériques de features.

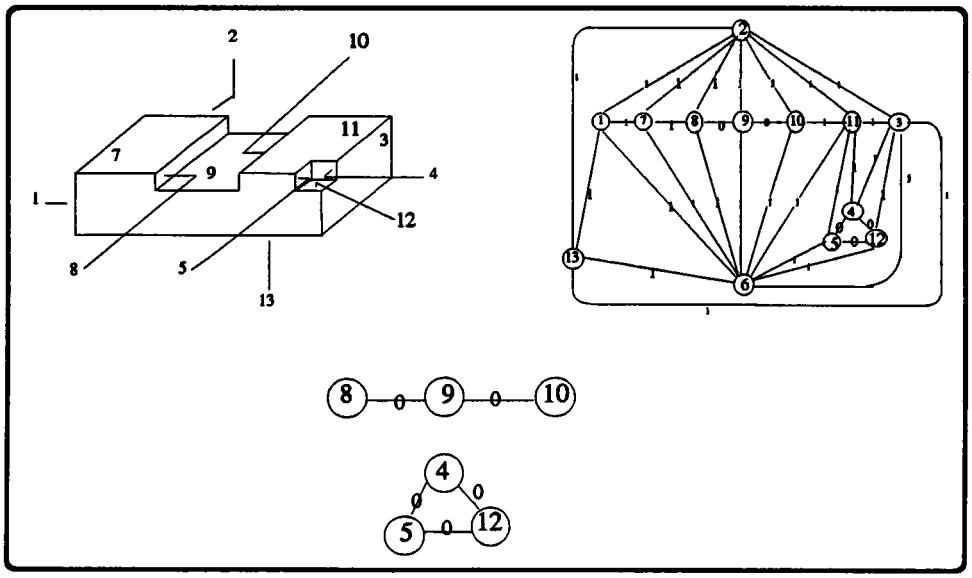


figure 4.5: graphe des adjacences (b) et graphe simplifié (c).

Cette méthode, dans l'état actuel d'avancement, présente tout de même des inconvénients de taille:

- la détection des trous ne fonctionne que partiellement, et rien n'est fait concernant les anses et ponts.
- l'imbrication de features n'est résolue que dans certains cas précis.
- l'identification d'une caractéristique de forme à l'aide de graphes génériques n'est pas absolument fiable: elle doit parfois s'appuyer sur des calculs géométriques supplémentaires et il arrive même que ses résultats soient discutables (fig 4.6).

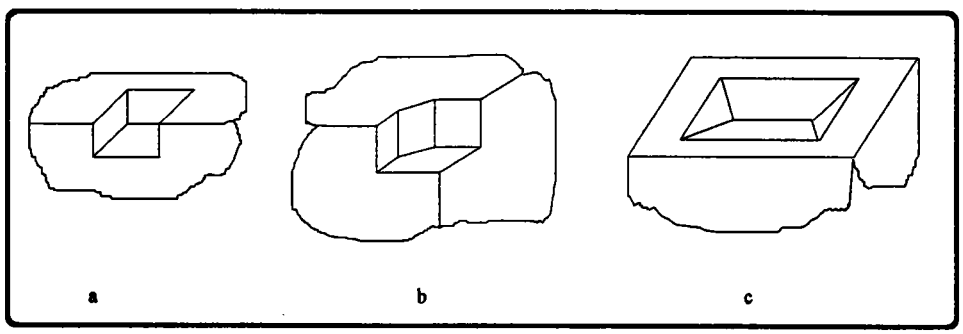


figure 4.6: les trois features (a=encoche borgne, b=marche, c=dépression) ont même graphe alors que leurs processus de fabrication sont différents (c doit généralement être précédé d'un perçage).

4.2.1.3. Extension à des facettes non planes

Nous évoquons à présent deux tentatives visant à la prise en compte de l'extraction de caractéristiques de forme s'appuyant sur des surfaces non planes, toujours dans le cas d'une modélisation par les limites:

- surfaces cylindriques, coniques, sphériques, toriques et circulaires pour la première [CHO 84],
- surfaces de degré 1 ou 2 pour la deuxième [CHU 90].

L'approche de Choi, Barash et Anderson [CHO 84] consiste principalement à repérer successivement:

- certaines encoches,
- certains trous,
- certaines poches.

Dans les trois cas, on cherche d'abord une face de départ, puis, en utilisant les relations d'adjacence du B-rep (liaisons face-face), les faces complétant le feature. Nous illustrons ceci en parcourant la méthode de détection d'un trou.

- la face de départ, F_1, doit être plane et comporter un contour intérieur circulaire.
- elle doit partager le contour circulaire avec une face F_2, du type de celles de la figure 4.7. Eventuellement, F_2 peut elle même être adjacente à une autre face de ce type, et ainsi de suite jusqu'à F_{n-1}.

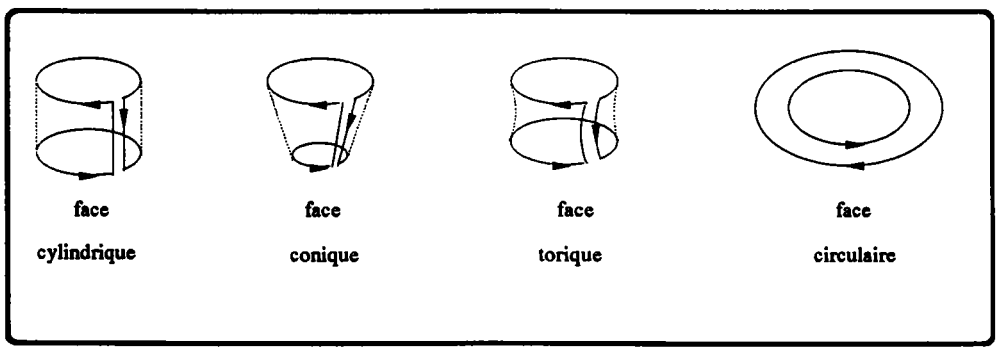


figure 4.7: faces autorisées dans un trou

- F_n, la dernière face du trou, peut être un cercle (trou à fond plat), un cone (trou à fond conique) ou une face plane comportant un contour intérieur circulaire (trou de part en part) (fig 4.8).

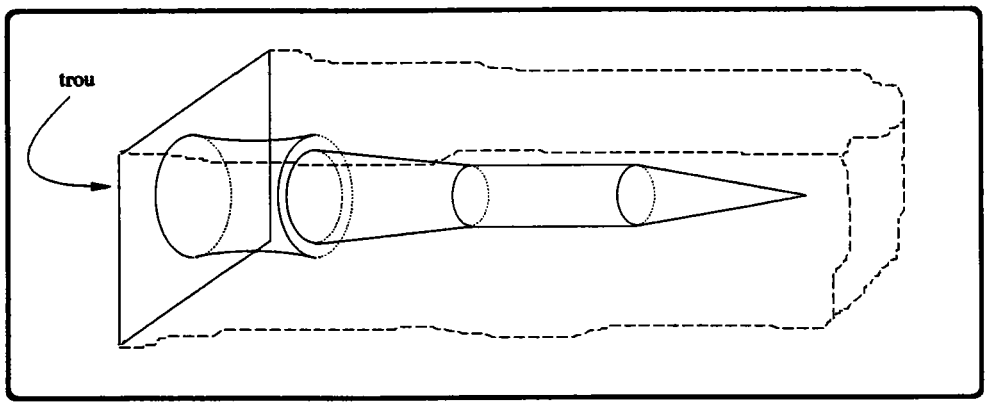


figure 4.8: un exemple de trou décelable par la méthode

Comme on peut le voir, la méthode est très ciblée et son extension semble devoir passer par une énumération de cas, non par une définition générique de l'objet trou.

Le même défaut se retrouve dans les travaux plus récents de Chuang et Henderson: ceux-ci consistent à établir un graphe dont les noeuds correspondent aux sommets de l'objet et les arcs à ses arêtes, autrement dit un modèle fil de fer, à cette différence que l'on ne stocke pas, dans un noeud, les coordonnées du sommet mais un "type de sommet", significatif du nombre, de l'ordre et de la nature (convexe, concave ou lisse) des arêtes et des faces qui le partagent (fig 4.9).

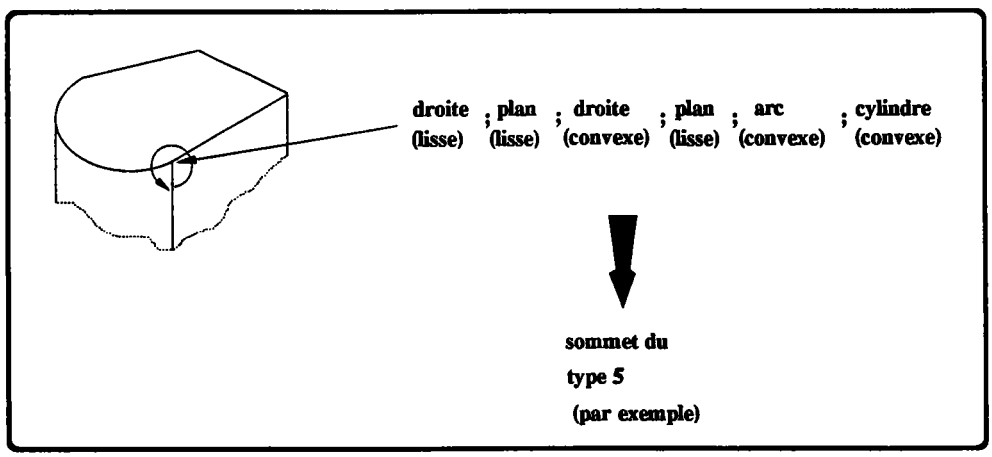


figure 4.9: la liste orientée des arêtes et des faces d'un sommet définit son type.

Il faut alors rechercher, dans le graphe de l'objet, les occurrences des sous-graphes associés aux caractéristiques de forme: mais alors que Joshi et Chang(4.2.1.2) parvenaient à définir des graphes génériques pour chaque type de caractéristique, il semble qu'il faille ici en définir autant qu'il y a de variations possibles dans le feature, d'où, pour être exhaustif, une grande quantité de cas.

Ceci implique des temps de calcul importants (la recherche des occurrences d'un sous-graphe est prohibitive) et des efforts considérables pour la définition et l'intégration de ces sous-graphes dans une base de connaissances.

Pour réduire ce deuxième inconvénient, les auteurs proposent qu'un opérateur crée une pièce contenant le nouveau feature, et indique interactivement au système d'extraction la partie intéressante: celui-ci peut alors automatiquement déduire le graphe de la caractéristique et l'intégrer à sa base de connaissances.

4.2.2. Méthodes associées aux représentations C.S.G.

La difficulté essentielle, avec une modélisation du type arbre de construction, vient de la non unicité de la représentation [WOO 88b]. Rappelons que cet état de fait se manifeste à deux niveaux:

- les propriétés des relations ensemblistes font qu'une même expression peut être formulée de différentes façons:

$$. (a \cup b) \cup c = a \cup (b \cup c)$$
$$. (a - b) - c = a - (b \cup c)$$

- d'un point de vue plus géométrique, on peut créer une même pièce à l'aide de plusieurs jeux différents de volumes primitifs.

La détection d'un feature, conséquence de la présence d'un certain nombre de primitives et d'un certain arrangement de ces dernières, est donc rendue d'autant plus délicate.

Aucun algorithme d'extraction n'est pleinement satisfaisant; nous donnons dans la suite les grandes lignes des travaux de Lee et Fu [LEE 87] d'une part, et de ceux de Perng, Chen et Li [PER 90] d'autre part. Ils ont en commun de chercher à restructurer l'arbre C.S.G. dans une organisation reflétant les méthodes de fabrication, autrement dit en y insérant autant d'opérateurs de différence que possible (fig 4.10).

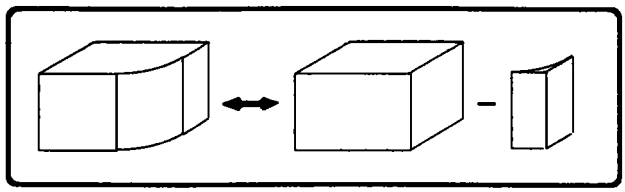


figure 4.10: la non unicité de la modélisation par arbre C.S.G. permet de donner d'une même pièce plusieurs représentations dont certaines facilitent la préparation à la fabrication (process planning).

Dans les deux cas, on supposera l'arbre dépourvu d'opérations d'intersection, ces dernières pouvant s'exprimer (en théorie tout au moins) en terme d'unions et de différences (A & B = A - (A - B)).

4.2.2.1. Travaux de Lee et Fu

La méthode de Lee et Fu tient en trois points:

- associer à chaque volume primitif un ou plusieurs segments orientés, appelés axes principaux, significatifs de sa taille et de son orientation.
  Déduire des positions relatives de ces axes la nature et les attributs géométriques des features.
- rassembler dans un unique sous-arbre les noeuds intervenant dans le feature.
- introduire dans ce sous-arbre des excès de matière dans des opérations de soustraction.

Nous explicitons ces trois points sur un exemple précis qui reprend celui des auteurs et discutons ensuite des limites de la méthode.

- *détection des features à l'aide d'axes principaux.*
  A un cylindre est associé un unique axe (fig 4.11.a).
  A une boîte sont associés douze axes, partant des sommets et qui ne sont rien d'autres que les arêtes (fig 4.11.b). A chaque sommet sont associés trois des axes, orientés de manière à ce que la boîte se trouve dans le premier octant du repère qu'ils définissent avec le sommet (fig 4.11.c).

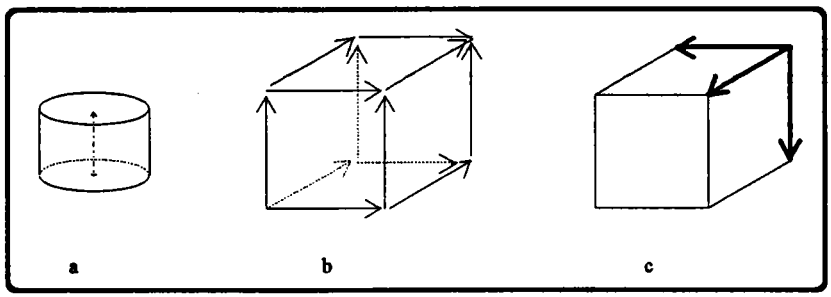


figure 4.11

En vérifiant:

- que les trois axes mis en évidence sur la figure 4.12 sont parallèles, de même longueur et de même "hauteur" (les segments joignant leurs origines leurs sont perpendiculaires),
- que l'axe du cylindre est à une distance égale au rayon des deux autres axes,
- que les trièdres situés aux extrémités des axes de la boîte sont superposables à l'aide d'une simple translation ,
- que l'axe du cylindre est convenablement situé par rapport à ces deux trièdres,

on détectera que la position du cylindre est telle qu'il crée un cylindre entre les deux boîtes.

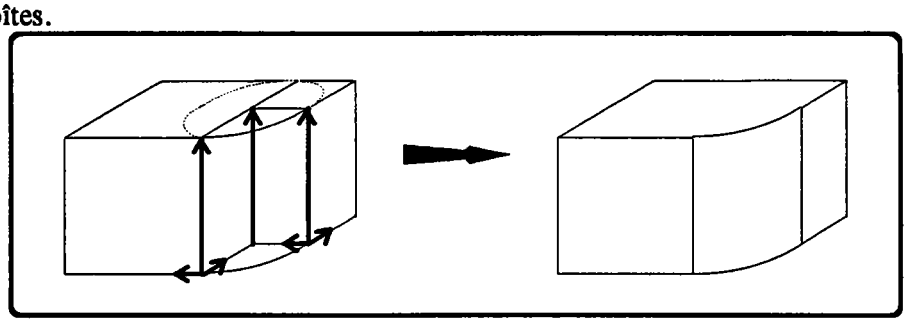


figure 4.12

- *rassemblement dans un sous-arbre des noeuds (un cylindre, deux boîtes) participant au feature.* Ceci est en partie réalisé à l'aide des propriétés ensemblistes (fig 4.13).

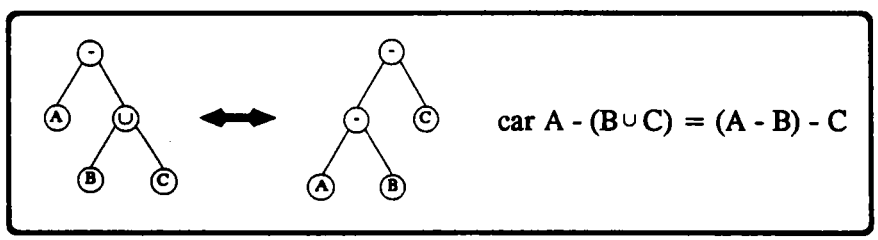


figure 4.13: on utilise une égalité ensembliste pour:

- descendre A d'un niveau
- monter C d'autant
- échanger B avec un de ses cousins.

- *réorganisation du sous-arbre*: jusqu'à présent, les traitements ont consisté à déplacer des noeuds; nous allons à présent en modifier certains en les remplaçant par un ou plusieurs autres, dans l'intention d'introduire des éléments dont le processus de fabrication est connu. Cette opération est compliquée; la figure 4.14 l'illustre en concluant sur l'exemple de l'arrondi.

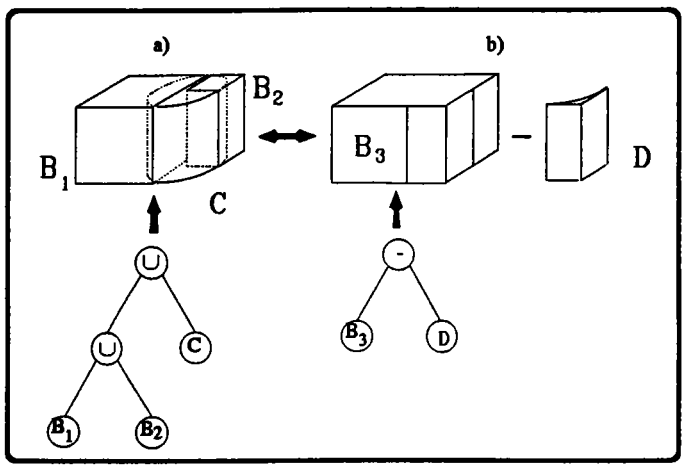


figure 4.14: en fusionnant les deux boîtes B1 et B2 (a), on peut réexprimer le feature sous la forme d'une différence (b), donc faire le parallèle avec un enlèvement de matière (usinage). Mais la chose est déja beaucoup moins simple si B1 et B2 ne peuvent être réunies en une seule boîte, ou si B1 participe à plusieurs features.

Cette méthode, illustrée par le cas précis de l'arrondi semble difficilement généralisable. En effet, les calculs à mener sur les axes principaux se compliquent considérablement dès qu'il s'agit de détecter d'autres catactéristiques (poches, marches, anses, trous...) et ne sont d'ailleurs pas évoqués dans [LEE 87]. De plus, il faut non seulement mettre en évidence un agencement d'axes caractéristique d'un feature, mais de plus vérifier que le feature n'est pas altéré par d'autres parties de l'arbre, ce qui est d'une grande complexité, spécialement si l'on ne raisonne que sur des axes comme le veut l'esprit de la méthode.

En revanche, l'arrangement de l'arbre selon certaines règles constitue un point de départ pour une réflexion visant à diminuer la diversité des représentations possibles d'un même objet, donc de la non-unicité du modèle C.S.G.

4.2.2.2. Travaux de Perng, Chen et Li

L'objectif est de transformer l'arbre C.S.G. en arbre D.S.G. (Destructive Solid Geometry: arbre ne contenant que des différences), de telle sorte que la feuille la plus à gauche corresponde au brut et que chaque sous-arbre droit se réduise à une primitive décrivant un excès de matière.

Les enlèvements successifs de tous ces excès au brut modélisent les opérations d'usinage aboutissant à la pièce.

Pour aboutir à la représentation D.S.G., les auteurs associent à chaque primitive une boîte englobante (décrivant un brut relatif) et le complémentaire de la primitive par rapport à l'englobant (décrivant l'excès de matière). Ainsi, à l'aide de techniques voisines de celles de Lee et Fu, mais en décomposant aussi parfois les primitives en mouvement dans l'arbre, ils parviennent à rassembler tous les englobants partiels en un brut de la pièce.

Cette méthode ne semble applicable qu'en respectant certaines hypothèses:
- des directions privilégiées pour les primitives,
- des contraintes sur le chevauchement des objets.

De plus, le brut obtenu est de taille importante, ce qui est sans doute non réaliste en fonction des coûts de fabrication.

4.3. Lien avec l'extrusion

Ce passage en revue de quelques techniques d'extraction de caractéristiques de forme montre que le champ des algorithmes dévolus aux modèles B-rep est encore largement supérieur à celui des C.S.G., d'autant plus que les coniques tendent à être prises en compte, ce qui réduit le gros désavantage des représentations uniquement polyédriques d'être des modèles approchés.

Il n'en reste pas moins que les arbres C.S.G. restent prometteurs du fait de la représentation concise et concentrée (dans un sous-arbre) qu'ils offrent des caractéristiques de forme, ce type d'agencement se prêtant bien aux modifications (déplacement, déformations... de la caractéristique).

Les méthodes se répartissent finalement en deux catégories: celles aboutissant à une décomposition de l'objet en composantes volumiques et celles mettant en évidence des caractéristiques correspondant à des enlèvements de matière.

Telles quelles, les premières sont relativement éloignées des opérations d'usinage: on met effectivement en avant certaines caractéristiques technologiques (dépressions) mais celles-ci sont tout-à-fait générales et il doit être difficile d'en déduire des procédés d'usinage, donc d'avoir une approche générative. Par contre, cette vocation de décomposer l'objet en composantes et la décomposition étant, sous certaines conditions, unique, font que la méthode se prête bien à la comparaison de pièces, en particulier sur des critères technologiques (deux dépressions imbriquées sur le dessus, un trou de part en part...). En cela, elle participe plutôt à l'esprit technologie de groupe, consistant à retrouver un objet déjà traité avec des caractéristiques identiques et à adapter son programme d'usinage à celui de la nouvelle pièce.

La décomposition étant structurée dans un graphe, on peut faire le parallèle avec un arbre C.S.G. dont les seules opérations seraient des collages. Un moyen de se rapprocher des techniques de fabrication et d'enlèvement de matière serait donc de restructurer cet arbre dans un format D.S.G. comme c'est actuellement le cas pour les arbres de construction. On trouvera dans [FAL 87] des éléments de réflexion sur ce thème.

Les méthodes qui mettent en évidence des enlèvements de matière (encoches, marches, perçages, congés...) se rapprochent davantage des techniques de fabrication et permettent d'envisager une approche générative, consistant à déduire de la pièce elle-même la suite des opérations qui la dégagent d'un brut. La presque totalité de ces excédents de matière consiste en objets prismatiques ou de révolution.

Les générateurs de ces objets extrudés ne sont toutefois pas utilisables tels quels car ils ne correspondent généralement pas à la forme d'un outil: il faut donc calculer la ou les trajectoires qui font que l'outil balaie un volume sensiblement égal au volume extrudé à usiner.

Il serait certainement intéressant d'essayer d'établir un lien entre la trajectoire de l'outil et le couple (*générateur,déplacement*) définissant l'objet extrudé. Il semble par exemple que dans certains cas, les passes successives de l'outil pourraient être des offsets de la trajectoire (pour un prisme généralisé, par exemple) ou du générateur (pour l'usinage d'une poche [SEO 90].

Ces méthodes mettant en évidence les enlèvements de matière peuvent elles-mêmes être réparties en deux catégories: celles visant à l'usinage de zones précises de la pièce,

correspondant à des caractéristiques, et celles prétendant à une approche globale, partant d'un brut qu'elles déterminent automatiquement et calculant la suite des usinages menant à la pièce désirée (approche D.S.G.).

La différence se fait essentiellement au niveau du brut, fourni par un intervenant extérieur dans le premier cas et calculé dans le deuxième. Ces deux approches se caractérisent donc, pour la première, par une perte de temps et d'avantage de charges pour l'opérateur et par une perte conséquente de matériau pour la deuxième.

Mais que l'on s'intéresse à l'une ou à l'autre, le problème reste posé d'associer à une extrusion donnée (une caractéristique) une autre extrusion, décrivant le déplacement d'un outil et théoriquement égale à la première: en clair, il s'agit donc de paramétrer les déplacements d'outils par les features extraits.

Ces remarques prennent toutes pour point de départ l'extraction de caractéristiques. Or, que l'on s'intéresse aux représentations B-rep ou C.S.G., aucun des algorithmes d'extraction n'est pleinement satisfaisant (soit qu'ils n'atteignent pas leurs objectifs, soit qu'ils soient trop ciblés) et l'on est loin de pouvoir se passer d'une intervention humaine. La solution consiste alors certainement à éviter cette étape calculatoire en proposant directement au concepteur des facilités d'intégration de caractéristiques dans la pièce (création d'un encoche, insertion d'un perçage...).

Se posent alors les problèmes d'utilisation interactive des caractéristiques: dans plusieurs publications, on propose par exemple qu'un opérateur dessine la caractéristique afin qu'elle soit enregistrée par le logiciel et ensuite réutilisable: on s'approche alors tout simplement de la notion de paramétrage par des notions non géométriques.

Se posent également les problèmes d'adéquation de la structure de données [REQ 86] [GOS 88] [MAS 90] (conservation de l'information technologique, outils associés...): il faut que les informations technologiques fournies par le concepteur figurent explicitement dans le modèle de façon à ce que l'on ne soit plus réduit, comme actuellement, à les détecter par des méthodes compliquées et pas toujours très satisfaisantes.

Poussé à l'extrême, ce raisonnement amène à limiter les outils du bureau d'étude en fonction des moyens de la fabrication: si le travail des autres étapes de la production s'en trouve facilité, la créativité du concepteur en est d'autant réduite: un tel choix doit donc être le fruit d'une décision longuement mûrie.

Notons enfin que toutes les difficultés ne s'envolent pas pour autant: les problèmes d'accessibilité à la caractéristique à usiner, le calcul de la trajectoire de l'outil et des interférences outil/objet, la simulation de l'usinage à des fins de vérification se posent encore avec autant d'acuité.

Pour illustrer cette approche, nous citons une proposition récente de Cavendish [CAV 90] permettant une modélisation mathématique de caractéristiques s'apparentant tout-à-fait à des emboutissages: elle consiste à déformer une surface implicite ($z = f(x,y)$) en la raccordant localement à des portions d'autres surfaces du même type, de façon à y insérer des poches et des protubérances.

Elle n'est évidemment pas adaptée à la réalisation de tuyaux, mais autorise la conception de surfaces dites fonctionnelles (panneaux intérieurs de voitures, d'avions...) qui apparaissent en bien plus grand nombre que les panneaux extérieurs (carrosseries) et qui ne peuvent, du fait de leurs formes irrégulières, être représentées efficacement de la même manière que ces dernières (par des Bézier, des splines,...).

Une orientation vers une conception partant d'un brut par enlèvements successifs de matière, correspondant en particulier à des extrusions, paraît difficile. En effet, certaines formes seraient alors impossibles à décrire dans la pratique et cela limiterait considérablement la créativité du concepteur. Une telle approche ne pourrait s'appliquer, comme c'est le cas de la méthode précédente, que pour des applications spécifiques.

L'objectif à long terme est cependant d'aller plus loin encore et d'arriver à stocker la totalité des informations relatives à un objet dans une unique structure centralisée de haut niveau, enrichie et exploitée par tous les intervenants dans le cycle de vie du produit (spécification, conception, fabrication, commercialisation..). On pourra alors parler de **modelisation produit** à part entière.

Cette démarche doit éviter de recalculer des informations devenues implicites ou d'en ré-acquérir certaines, déjà fournies à un autre stade mais oubliées depuis, comme c'est souvent le cas du fait de l'automatisation des divers îlots de la production sans réel soucis des besoins de l'aval ou de l'amont.

On pense ainsi arriver à intégrer l'ensemble de la production mieux que cela ne se ferait avec des procédés d'interfaçage ou de normalisation [MON 90]:

- une interface traduisant une structure d'un format dans un autre, elle est nécessaire dès qu'il y a transfert d'informations entre deux secteurs. L'inconvénient vient alors du fait que lorsqu'une application se crée, il faut établir autant de nouvelles interfaces qu'elle a d'interlocuteurs, d'où une multiplication des interfaces et des problèmes attenants (maintenance).
- les normes (SET, IGES) ont, elles, le défaut de n'intégrer les notions nouvelles que longtemps après leur avènement: à l'heure actuelle, rien n'est encore établi, en particulier, en matière de feature, ce qui impose, pour présenter l'information d'une

manière indépendante des applications, de la décomposer en éléments de bas niveau, souvent de nature géométrique et, donc, à faible teneur sémantique.

CONCLUSION

CONCLUSION

Notre objectif a été au fil de ce mémoire d'exposer les possibilités de création et de modélisation de la fonction d'extrusion. Cette démarche a d'abord exigé un effort de formalisation car, dans la mesure où l'on ne retrouve pas cette formalisation dans la littérature, nous pensons que cet apport théorique était indispensable avant de s'attacher à des points particuliers: il a donc fait l'objet du premier chapitre.

Nous nous sommes attachés dans le deuxième chapitre à montrer les différentes manières de modéliser un objet extrudé. Nous avons tenté de définir des algorithmes profitant au mieux des particularités de ces modèles. L'intersection par un rayon nous paraît être une opération significative dans la mesure où elle peut être utilisée dans diverses applications telles la combinaison booléenne d'objets, le tracé de rayons ou l'identification.

Le fait de pouvoir mettre en oeuvre un tel calcul sans évaluation par les frontières peut permettre de réaliser un outil de base performant et fiable.

Il nous a également paru important d'étudier d'une manière la plus exhaustive possible les différentes façons d'intégrer un modèle d'extrusion dans un C.S.G. ou un B-rep.

Pour nous pencher plus avant sur certains problèmes ciblés, nous nous sommes intéressés dans un troisième chapitre à deux syntaxes particulières de la fonction d'extrusion: le prisme généralisé et l'objet de révolution qui ont respectivement soulevé le problème de la combinaison de contours coplanaires et de l'intérêt des opérateurs d'Euler.

On peut regretter que d'autres aspects pratiques n'aient été programmés. La raison en est que les diverses facettes de la fonction d'extrusion sont très variées et que leur mise en oeuvre exige un matériel de base important, souvent inédit. Or l'équipe travaillant sur le projet SACADO n'a pas utilisé un logiciel existant pour la mise en pratique de ses idées et a donc dû consacrer une grosse partie de son temps de développement à ces fonctions de base, notamment pour la version 2D qui est relativement aboutie.

Ce choix de ne pas adopter un logiciel existant a été motivé par une volonté d'utiliser, tout particulièrement pour la modélisation du dialogue, des outils originaux et efficaces satisfaisant aux principes exposés dans [GAR 88].

Volontairement, nous n'avons fait état ni des difficultés soulevées par la réalisation des fonctions de base indispensables, ni de celles, plus spécifiques, rencontrées dans le cadre d'une implantation sur micro-ordinateurs, essentiellement imputables au faible volume mémoire disponible (cet inconvénient disparaît sur les systèmes récents avec l'apparition d'utilitaires gèrant de grosses quantités de mémoire).

Globalement, on constate que la fonction d'extrusion constitue un outil satisfaisant pour la réalisation de formes qui ne sont pas trop complexes car plus le niveau de détail est important (variations du générateur, complexité du déplacement), plus le dialogue et l'algorithme de modélisation se compliquent. L'extrusion semble donc toute indiquée pour l'établissement de maquettes ou de silhouettes sans un complet souci de réalisme. Cette remarque s'applique en fait à n'importe quel opérateur de création de forme mais elle prend ici une valeur particulière car d'un point de vue théorique, toute forme peut être exactement décrite par une extrusion.

Les développements évoqués dans le troisième chapitre sont intégrés dans la version 3D de SACADO. Celle-ci est bien moins agréable à l'utilisation que ne l'est la version 2D. Dans l'immédiat, nous allons donc chercher à améliorer ceci en perfectionnant le dialogue selon les principes utilisés dans [GAR 88] et en optimisant certains algorithmes (dont l'élimination des faces cachées) afin d'améliorer les temps de réponse. Nous pourrons alors facilement intégrer 2D et 3D dans un tout cohérent et d'une certaine efficacité.

L'étape suivante se démarquera assez nettement de la précédente: nous avons vu dans le quatrième chapitre la nécessité d'incorporer dans le modèle d'autres informations que celles purement géométriques, dont la portée est limitée à la description de la forme des objets. L'actuel modèle sera donc modifié de manière à intégrer des informations technologiques et à se rapprocher du concept de *modèle produit*. Il est encore prématuré d'en donner une description mais il est probable qu'il consistera en un arbre de construction (pour les aspects "modélisation de l'historique" et "paramétrisation de la forme par des composantes -volumes primitifs ou features-), complété d'un B-rep (pour les traitements plus en rapport avec les surfaces -tolérancement, précision, usinage-).

Théoriquement, et nous espérons ne pas nous tromper sur ce point, ce changement de la structure du modèle ne devrait pas (ou peu) affecter ce qui aura été réalisé jusqu'alors car nous pensons avoir bien détaché les couches application de la couche modélisation (cf 3.3).

ANNEXE I

ANNEXE 1 : le modèle 3D

Le modèle géométrique 3D utilisé dans SACADO est un très classique B-rep polyédrique de type faces-arêtes-sommets, doté de quelques structures complémentaires.

Il est constitué de sept tables dont nous décrivons sommairement les éléments.

1) table des objets 3D (polyèdres). Un objet est caractérisé par:

* *nature*: il s'agit d'un code précisant la nature du polyèdre: quelconque, objet de révolution, prisme ou prisme généralisé.
* *historique*: liste des éléments utiles pour une reconstruction; généralement, il s'agit de numéros de contours 3D de la sixième table.
* *première et dernière faces* du volume: ceci permet un rapide parcours des faces qui sont chaînées entre elles.
* *premiers et derniers sommets et arêtes*, pour un rapide parcours des sommets et des arêtes du volume. Par exemple, on calcule ainsi facilement un englobant de l'objet.

2) table des faces et des contours.

Un élément de cette table est susceptible de décrire une face ou un contour.

Dans le premier cas, il décrit en fait le contour extérieur de la face, si bien que tout élément décrit finalement un contour (extérieur ou intérieur). Ses caractéristiques sont:

* le numéro du vecteur normal au contour
* le numéro de la première arête du contour
* si le contour est extérieur, le numéro du premier contour intérieur (0 à défaut)
* si le contour est intérieur, le numéro du contour intérieur suivant s'il existe, celui du contour extérieur sinon: ceci permet un chainage dans une liste circulaire des contours d'une face.

3) table des arêtes: chaque arête comporte:

* deux numéros de points (sommets extrémités de l'arête)
* ces deux points définissent un sens de parcours intrinsèque de l'arête. On nomme contour gauche (CG) le contour empruntant l'arête dans ce sens et contour droit

(CD) celui qui l'emprunte à contre-sens (rappel: toute arête appartient exactement à deux contours sur lesquels elle est parcourue dans des sens opposés).
On retrouve donc les deux faces partageant une arête en consultant les contours gauche et droit de l'arête et en exploitant le fait que les contours d'une face sont chainés dans une liste circulaire.

* SFG et SFD (Suivant Face Gauche ou Droite) sont les numéros des arêtes suivant l'arête courante sur les contours gauche et droit. La première part, sur CG, de l'extrémité de l'arête courante (au sens de l'orientation intrinsèque) alors que la deuxième, SFD, part de son origine.
* ar_suiv est le numéro de l'arête suivant l'arête courante dans le volume: toutes les arêtes sont donc triplement chainées: elles ont une suivante sur chaque contour adjacent et une autre sur le volume.

4) table des points: chaque point comprend un triplet de coordonnées et le numéro de l'objet auquel il appartient.

5) table des vecteurs: un vecteur est un triplet de coordonnées.

6) table des contours 3D: les créations de volumes s'appuient toutes, dans notre approche, sur des contours (parfois ouverts). Cette table établit un lien entre le modèle 3D et le modèle 2D. Pour chaque contour 3D, on a:
   * un numéro de contour 2D, soit une entrée dans le modèle 2D.
   * le numéro de la transformation géométrique qui amène le contour 2D sur le contour 3D.
   * les quatre coefficients de son plan; les trois premiers constituent la normale orientée normée du contour.

7) table des transformations géométriques
   Une transformation est un couple (matrice,historique): la matrice est de dimension 4x4 et l'historique décrit évidemment l'historique de la transformation: on saura par exemple que l'une d'elles est la composée d'une rotation par rapport à (Ox) de 30° puis d'une translation d'un vecteur V puis encore d'une rotation autour de (Oy) de 25°.
   L'historique est fort utile pour le calcul de la matrice inverse ou pour d'éventuelles simplifications lors de la composition: avant de multiplier les deux matrices, on compare les historiques et s'il y a possibilité de simplification (deux transformations consécutives inverses qui s'annulent, ou deux transformations de même nature, que l'on combine en une seule), la matrice composée est évaluée à partir de l'historique.

ANNEXE II

ANNEXE 2 : algorithme de tri de contours

Cette annexe décrit un algorithme simple ordonnant un certain nombre de contours coplanaires selon leurs inclusions (fig A.1.a) et en déduisant un certain nombre de faces composées d'un contour extérieur et d'éventuels contours intérieurs fig (A.1.c).

L'hypothèse est faite que les contours ne se coupent pas: la position par rapport à un contour C_2 (dedans, dehors) d'un quelconque point d'un autre contour C_1 est donc significative d'une inclusion de C_1 dans C_2.

La méthode consiste à établir un arbre n-aire *arbre* dont les noeuds correspondent aux contours; l'arbre est ordonné de telle manière qu'un contour soit inclus dans son ascendant et contienne ses fils (fig A.1.b).

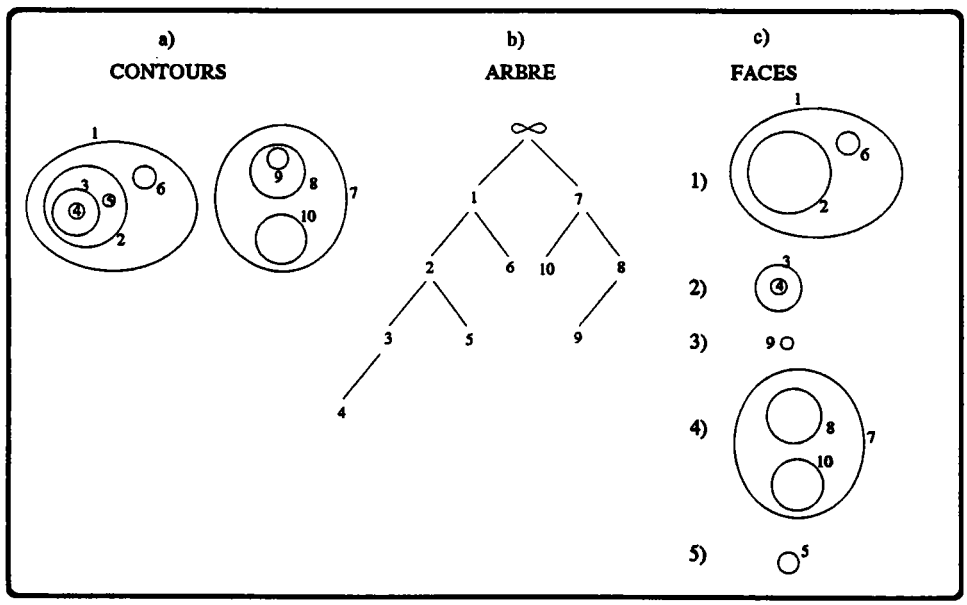


figure A.1

Pour rendre la démarche plus systématique et en particulier éviter d'avoir plusieurs racines, on supposera que l'arbre contient initialement un unique contour représentant l'espace entier et constituant donc un englobant global.

Un contour *C* est inséré dans l'arbre *arbre* en raisonnant récursivement:

. par hypothèse,*C* est inclus dans le contour du noeud *arbre* (on allègera la terminologie en considérant alternativement un noeud comme la racine d'un arbre ou un contour). Il doit donc figurer dans sa descendance.

. si *C* ne contient aucun des fils de *arbre*, il se peut qu'il appartienne à l'un des fils. Deux situations sont alors possibles: soit il appartient effectivement à l'un des fils et il doit être inséré dans le sous-arbre du fils, soit ce n'est pas le cas et il devient un nouveau fils de *arbre*.

Si par contre C contient des fils de *arbre*, ceux-ci deviennent alors ses propres fils et *C* devient un fils de *arbre*.

Les faces sont facilement déduites de l'arbre: chaque noeud d'un niveau impair de profondeur (la racine est le niveau 0) définit le contour extérieur d'une face et ses fils les contours intérieurs.

ALGORITHME

DONNEES : Une liste *LC* de contours coplanaires sans intersections.

RESULTATS : Une liste *LF* de faces.

DEBUT

. Créer un arbre *arbre* ne contenant qu'un noeud correspondant au contour infini.

. **{ initialisation de l'arbre des inclusions }**

pour chaque contour *C* de *LC* **faire**

classer_contour (*C*,*arbre*)

fpour

. **{ création de la liste *LF* des faces }**

LF = liste vide

créer_faces (*arbre*,*LF*)

FIN

MODULE CLASSER-CONTOUR (cont,arbre)

{ Insère le contour *cont* à sa bonne place dans l'arbre *arbre* dont la racine EST SUPPOSEE ETRE UN ENGLOBANT DE *cont* }

DEBUT

{ Etablit une liste *l_fils* des fils de *arbre* contenus dans *cont* }

l_fils = liste vide

pour chaque fils *F* de *arbre* **faire**

si le premier point de *F* est dans *cont* **alors** **{ *F* est dans *cont* }**

ajouter *F* à *l_fils*

fsi

fpour

si *l_fils* est non vide **alors**

{ *cont* est inclus dans *arbre* mais contient certains de ses fils }

transférer les contours de *l_fils* de la liste des fils de *arbre* dans la liste des fils de *cont*

faire de *cont* un fils de *arbre*

sinon

{ recherche un fils de *F* contenant *cont* }

contenu = faux ; *P* = premier point du contour *cont* ;

pour chaque fils *F* de *arbre* et **tant que** non *contenu* **faire**

si *p* est dans *F* **alors** *contenu* = vrai **fsi**

fpour

si *contenu* **alors** **{ succès dans la recherche: *f* contient *cont* }**

{ *cont* étant dans F, il doit être classé par rapport au sous-arbre dont *F* est la racine}

classer_contour (*cont*,*F*)

sinon

{ *cont* est contenu dans *arbre* mais pas dans un de ses fils}

cont devient un nouveau fils de *arbre*

fsi

fsi

FIN

MODULE CREER-FACES (arbre,LF)

{ Crée une liste *LF* de faces }

DEBUT

pour chaque fils *F* de *arbre* **faire**

{ Crée une liste *LC* des contours d'une face avec, en tête, le contour extérieur puis les contours intérieurs }

LC = (*F*) **{ *F* est le contour extérieur de la nouvelle face }**

pour chaque fils *FF* de *F* **faire**

créer_faces (*FF*,*LF*) **{ Crée d'abord les faces les plus internes }**

ajouter *FF* en queue de *LC* **{ Ajoute un contour intérieur à la face courante }**

fpour

ajouter à *LF* une face constituée des contours de *LC*

fpour

FIN

ANNEXE III

ANNEXE 3: la convivialité dans SACADO

Afin de bien situer la manière dont sont perçues dans SACADO les notions de dialogue et d'interactivité, nous insérons la copie d'un article paru en 1988 dans les actes du congrès MICAD (pages suivantes).

UNE APPROCHE NOUVELLE DE LA CONVIVIALITE DANS UN SYSTEME DE CAO: LES PRINCIPES DU DIALOGUE DANS SACADO

Y. Gardan, J.P. Jung, J.N. Kolopp, C. Minich, W. Totino
Laboratoire de Recherche en Informatique de Metz
équipe CFAO et IA.

Résumé :

Le dialogue est l'un des aspects importants d'un système de CFAO, dans la mesure où il rend plus ou moins facile l'accès aux fonctionnalités du système. Nous proposons une approche différente des méthodes mises en oeuvre habituellement dans les systèmes de CAO, en montrant les principaux aspects dialogue de SACADO, Système Adaptatif de Conception Assistée et de Développement par Ordinateur.

Ce projet bénéficie d' une aide ANVAR (région Lorraine), de la collaboration et d'une aide de la société ARM Conseil, ainsi que d'une bourse CIFRE.

I. INTRODUCTION

On peut considérer que tout système de CAO moderne doit chercher, outre les aspects techniques classiques (modèle, outils de calcul ...), à être ouvert et convivial. Parmi ces objectifs, on s'attache dans cet article à l'aspect convivialité : il faut obtenir un dialogue souple, avec un minimum de contraintes pour l'utilisateur final (menus dynamiques, multifenêtrage ...) et un maximum de facilités pour construire, sans appel à un langage informatique, un dialogue adapté à un utilisateur et une application donnée (macro-menus, définition interactive d'icônes, d'un dialogue ...). Pour montrer notre approche du dialogue, nous nous appuierons sur le système SACADO, en cours de développement dans notre laboratoire, et dont la première maquette est opérationnelle [SAC 87]. Il est évident que ce système dispose d'un certain nombre de fonctionnalités que nous ne décrirons pas (modélisation, ouverture ...) [GAR 86].

Pour présenter notre approche nous décrivons :

- Le dialogue : ce dialogue est fondé sur sur un certain nombre de principes généraux. En particulier, notre approche est trés différente des approches fondées sur des logiciels de base répondant, par exemple, aux descriptions GKS ou PHIGS. Nous considérons, en effet, que tout en conservant une volonté d'indépendance en fonction des matériels, une mise en oeuvre du dialogue adaptée à l'application est plus performante et facilite de nombreuses opérations.

- Des exemples : à travers quelques opérations simples (constructions sous contraintes, contours, cotations), nous montrons l'interêt de notre approche. Nous présentons des fonctionnalités facilitant les actions de l'utilisateur (propositions de solutions, affichage élastique, changement de contexte ...).

II. LA GESTION DU DIALOGUE

II.1. Introduction

Nous nous fixons deux objectifs qui nous semblent fondamentaux :

- fournir des principes de dialogue dont la généralité permet :

  + de rester indépendant des applications.

  + d'imposer le minimum de contraintes à un utilisateur.

  + de ne pas limiter un utilisateur dans ses intentions, mais au contraire de rester en accord avec sa logique d'utilisation.

- faciliter la mise en oeuvre d'une application par :

  + la possibilité de définir interactivement un environnement de dialogue dynamique adapté à l'application en question.

  + la mise à disposition de primitives de dialogue utilisables facilement dans la programmation des traitements.

Le premier point nous a conduit à introduire des notions nouvelles qui sont définies dans le paragraphe suivant.

Le second point a permis de définir l'architecture logicielle du système qui fait l'objet du paragraphe II.3

II.2. Principes de bases

De façon classique, un tel système fonctionne par le choix de menus associés à des actions à entreprendre.

Dans ce cadre, le développement de la maquette 2D de SACADO a permis de mettre en évidence des fonctionnalités générales souhaitables s'appuyant sur le fait que le système doit être capable :

- de tenir compte du contexte d'utilisation.

- d'interpréter au mieux les intentions de l'utilisateur (en fonction de l'application).

Nous avons considéré que quelle que soit l'application :

- les interventions successives de l'utilisateur font passer le système d'un "contexte" (état) à un autre.

- dans un contexte donné, un menu est "actif" (visible) si on peut le sélectionner, et "non actif" (invisible) dans le cas contraire.

- plusieurs menus peuvent être liés à une même notion et ainsi faire partie d'une famille. Ils sont dits "SOUS-MENUS" d'un "PERE" (ex. : différentes façons de créer un segment).

- un contexte peut être momentanément suspendu par le choix d'un menu compatible. Celui-ci est dit "IMMEDIAT" car le traitement qui lui est associé est alors immédiatement lancé (sans pour autant interrompre le contexte en cours). On entre dans un contexte local au contexte en cours, à la sortie du contexte local on se retrouve au même point du contexte suspendu
  (ex. : + demande de création d'un segment lors de la création d'un contour

  + demande de zoom lors d'un effacement).

- un contexte peut être brutalement interrompu par le choix d'un menu incompatible. Celui-ci est dit "DIFFERE" car l'exécution de son traitement associé est différée dans un contexte pour lequel le menu est compatible
  (ex. : + demande de cotation lors de la création d'un contour).

- un contexte peut être précisé par le choix d'un menu qui n'a de sens que par rapport au contexte en cours. Le menu est dit "LOCAL" car il n'est accessible que dans le contexte en cours (activé en entrant dans le contexte et désactivé en sortant). Le traitement qui lui est associé est lancé immédiatement, lorsqu'il se termine le contexte suspendu reprend au point suivant l'interruption
  (ex. : lors d'un effacement, préciser s'il s'agit
  + de la totalité d'un objet
  ou
  + d'une portion d'objet).

Remarque : Des "SOUS-MENUS" sont associés à un menu "PERE", alors que des menus "LOCAUX" sont associés à un traitement terminal, en fait, ils sont assimilables à des touches de fonctions.

La suite de l'article donne des exemples pratiques qui devraient permettre de mieux comprendre l'intérêt de ces notions.

Le paragraphe suivant montre comment ces notions sont prises en compte dans SACADO.

II.3. Architecture logicielle

II.3.1. Le noyau du système

La plupart des systèmes distinguent les primitives de dialogue de façon fonctionnelle. Par exemple, dans GKS [END 84], on trouve quatre primitives de base :

- la sélection, dont la fonction est de permettre le choix d'une option menu.
- la désignation, dont la fonction est de permettre l'identification d'un objet.
- la localisation, dont la fonction est de permettre une récupération de coordonnées.
- la valuation, dont la fonction est de permettre l'attribution d'une valeur à un scalaire.

Une telle approche présente des inconvénients non négligeables.

D'une part, le fait même qu'il y ait plusieurs primitives ne facilite pas leur utilisation et nécessite de bien comprendre la fonction de chacune.

D'autre part, cela limite considérablement les possibilités offertes. En effet, l'appel de l'une quelconque de ces primitives place l'utilisateur dans un contexte figé en ne lui autorisant rien d'autre que la fonctionnalité prévue par la primitive en question.

De plus, il est essentiel de prendre en compte les notions de base définies au paragraphe précédent. Et ce, de façon automatique et transparente pour un programmeur.

Ceci nous a amené à ne considérer qu'une unique primitive de dialogue que nous appellerons "INTERACTION" (avec le système) et à articuler l'ensemble du système autour d'un MONITEUR de dialogue.

Outre la réalisation des principes de base, la primitive "INTERACTION" restitue des informations (coordonnées d'un point, objets proches du point, valeur d'un scalaire...) que le programmeur utilise selon ses besoins.

Le MONITEUR est lui-même un traitement utilisant la primitive "INTERACTION". Il s'appuie sur une modélisation des choix possibles (et des traitements associés) par [TOT 86]:

- une structure de données contenant des informations graphiques (représentation et position des menus) et des informations fonctionnelles (état [actif ou non] des menus,liens entre certains menus [sous-menus, différés, immédiats, locaux], actions à entreprendre, contexte en cours).

- une primitive d'appel au travers de laquelle sont lancés indirectement les traitements (interprétation).

L'état de la structure évolue avec le contexte d'utilisation. Ecrire une application devient équivalent à:

- écrire les traitements relatifs à l'application (avec une utilisation éventuelle de la primitive "INTERACTION").

- permettre la prise en compte de l'ensemble des traitements par la primitive d'appel (automatique).

- initialiser la structure de donnée en fonction d'un environnement de dialogue choisi par l'utilisateur.

Il suffit dès lors de lancer le MONITEUR de dialogue pour utiliser l'application.

II.3.2. Une application particulière

Afin d'optimiser la convivialité et l'ouverture du système, il apparaît comme crucial de permettre l'initialisation (et la modification) interactive de la structure définissant l'environnement de dialogue.

L'exploitation de l'aspect dynamique de la structure permet de résoudre le problème. Nous avons écrit une application dont les traitements sont spécialisés dans la création (et la modification) d'un environnement de dialogue relatif à une autre application.

Nous insistons sur le fait que cette application particulière fonctionne suivant les mêmes principes (et uniquement ceux-ci) qu'une autre application. Ce point est fondamental car il permet d'intégrer cette application comme fonctionnalité minimum de toute autre application (puisque toutes deux sont gérées par le MONITEUR).

L'utilisateur peut ainsi agir sur son environnement (sans quitter son application) en :

- définissant l'application par touches successives.
- ajoutant des options menus dont les traitements sont connus de la primitive d'appel du MONITEUR.
- supprimant des options devenues inutiles.
- créant des macro-menus (séquences de traitements) correspondant à une démarche qu'il a l'habitude de suivre.

III. EXEMPLES

III.1. les constructions sous contraintes

Une construction sous contraintes est définie par:

- le type d'objet à construire (point, segment, arc, cercle),

- des contraintes principales (tangent à, passant par, distant de...),

- les objets sur lesquels portent les contraintes principales,

- des contraintes secondaires permettant de choisir un objet parmi plusieurs (par exemple, un cercle tangent à deux cercles de rayon donné, peut être choisi parmi au plus 8 cercles candidats).

L'objectif a été de prendre en compte ces contraintes en utilisant les menus dynamiques et en minimisant le nombre de contraintes secondaires à préciser, ce qui rend le dialogue beaucoup plus simple que dans les solutions traditionnelles.

Le dialogue se déroule de la manière suivante:

- l'opérateur choisit par menu le type d'objet à construire,

- le sous-menu des constructions principales associé au type d'objet choisi apparait, et l'opérateur peut choisir l'une des constructions,

- en fonction de la construction demandée, le système demande des identifications ou des coordonnées,

- à partir des renseignements donnés, le système calcule les solutions répondant aux contraintes principales , recherche la solution la plus plausible, qui devient la solution courante et que nous appelons pour l'explication OBJET (voir exemple ci-dessous) , et la propose (l'affiche). En situation de PROPOSITION DE SOLUTION, les cas suivants peuvent se produire:

- si l'utilisateur identifie un nouvel objet(ou donne de nouvelles coordonnées, en fonction de la contrainte principale en cours), OBJET est créé dans le modèle, et le système traite une nouvelle construction du même type,

- si l'utilisateur montre le menu SUIVANT, le système choisit une autre solution plausible pour OBJET et la propose (l'affiche). Il se replace en situation de PROPOSITION DE SOLUTION,

- si l'utilisateur montre le menu ABANDON, le système abandonne la solution (rien n'est créé dans le modèle) et attend une nouvelle construction du type de celle qui était en cours,

- si l'utilisateur montre un autre menu, autre que certains menus à action immédiate(zoom par exemple), l'OBJET proposé est adopté et créé dans le modèle.

Considérons deux cas de construction sous contraintes: création d'un segment tangent à deux cercles et création d'un segment passant par deux points.

a) segment tangent à deux cercles:

L'opérateur a précisé qu'il veut créer un segment, tangent à deux cercles, et il a désigné deux cercles. La primitive d'INTERACTION, outre le nom (dans le modèle) des objets cercles montrés, retourne les coordonnées du point qui a servi à l'identification (Cf figure 1).

A partir de ces éléments, le système peut récupérer les paramètres des deux cercles (centres et rayons) dans le modèle et calculer les tangentes possibles(figure 2). A partir des deux points ayant servi à l'identification, il choisira la solution indiquée sur la figure 3, qui semble la plus plausible. Si l'utilisateur n'est pas satisfait de cette solution, il pourra demander l'une après l'autre l'affichage de toutes les solutions, jusqu'à satisfaction ou abandon (on boucle sur les solutions).

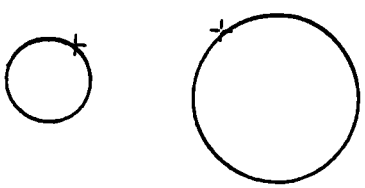


Figure 1

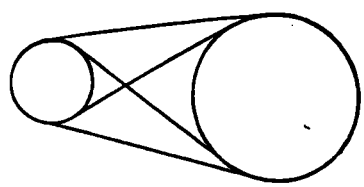


Figure 2

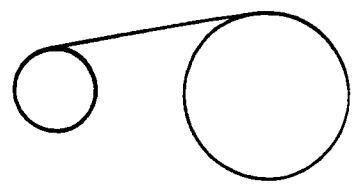


Figure 3

b) segment passant par deux points:

Il est très facile d'autoriser la définition des extrémités, non seulement par des coordonnées, mais également par n'importe quel type de construction sous contrainte. Il suffit de déclarer les menus correspondants à des constructions de points sous contraintes comme locaux. A chaque appel à la primitive INTERACTION, l'utilisateur peut sélectionner la contrainte de son choix (par exemple MILIEU d'un segment existant) et obtenir le point cherché.

c) ZOOM:

Il est possible à tout moment dans une construction d'agrandir une partie de la scène, sans abandonner la construction en cours. Il suffit de déclarer le menu ZOOM comme immédiat pour toutes les constructions sous contraintes.

III.2. Dialogue pour les contours.

III.2.1. Les contours évidents.

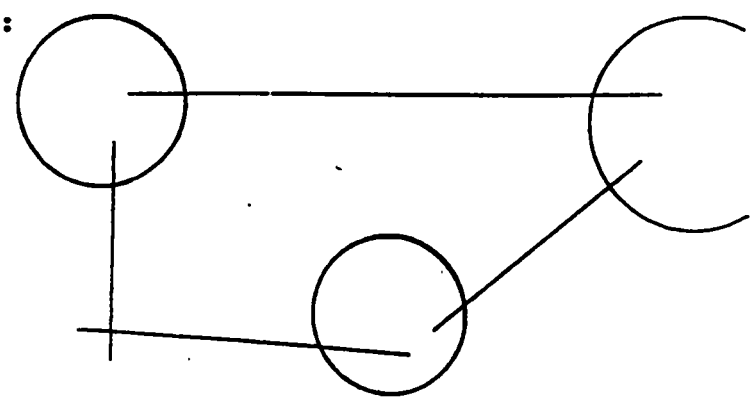
Cette forme de construction permet dans certaines conditions de réaliser un contour en un minimum d'interactions, soit, dans le cas présent en désignant un segment du contour et le sens de parcours de celui-ci. Il faut, pour cela, que la suite d'objets sur laquelle s'appuie le contour, réponde aux deux seuls critères suivants :

- chaque objet ne doit être coupé que par deux autres objets visibles.

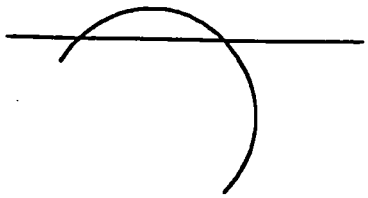
- il n'y a pas d'intersections multiples.

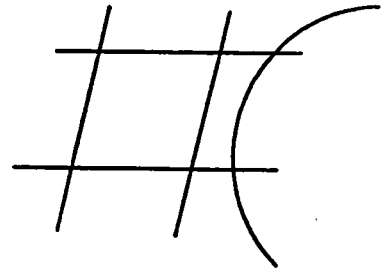
Nous avons jugé cette option fort utile dans la mesure où la personne établirait sa scène dans le but précis de construire son contour, celle-ci se rapprochant donc du contour voulu.

scène type :



situations interdites :





Reste le problème lié à l'orientation des cercles : ceux-ci sont parcourus dans le sens trigonométrique, donc selon le sens de parcours choisi au départ on obtiendra soit la figure 1, soit la figure 2.

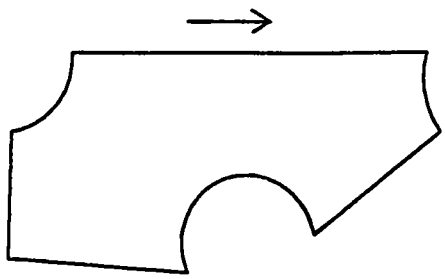


figure 1

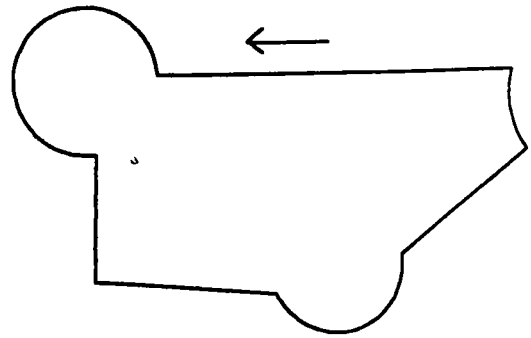


figure 2

Un menu permet, le cas échéant, d'inverser le sens de parcours d'un ou plusieurs cercles par simple désignation.

III.2.2. Les contours généraux.

Nous avons cherché, de même que précédemment, le moyen le plus simple et le plus naturel de désigner un ensemble d'objets afin de construire n'importe quel contour. Cette désignation est faite à l'aide de la primitive "INTERACTION".

Les difficultés sont liées au problème de spécification du sens de parcours des éléments et du choix de l'intersection d'un élément avec son suivant en cas d'ambiguïté. Une stratégie simple permet de réaliser n'importe quel contour en ne désignant, dans la plupart des cas, qu'une seule fois un objet, et au pire deux fois.

L'utilisation de la primitive "INTERACTION" pour la désignation et la spécification de certains menus comme "IMMEDIAT" de CONTOUR, fournissent à l'utilisateur des outils puissants et pratiques.

On peut donc, lors de la désignation des objets du contour, avoir accès à différentes fonctions sans quitter l'opération d'entrée des données.

Exemples :

- le zoom
Lors de la saisie, certaines parties du dessin peuvent être confuses et ne pas permettre une désignation précise. L'utilisateur peut alors abandonner momentanément l'opération en cours, agrandir la portion désirée, poursuivre sa saisie, puis revenir à l'espace de base afin de finir son opération de désignation.

- les constructions sous contraintes (III.1.)
En cours de saisie, l'utilisateur peut, à tout moment, interrompre celle-ci afin de rajouter un éventuel élément. Cette construction sera, en tous points, identique à celle faite dans d'autres conditions. L'objet ainsi construit pourra alors être intégré au contour au même titre que ceux déjà existants. L'opération terminée,la saisie reprendra au niveau du dernier objet désigné par l'utilisateur.

III.3 Dialogue de la cotation

Dans la maquette 2D de SACADO , les cotations portent sur un ou deux objets simples : points , segments , cercles et arcs .

Elles ont fait l'objet d'un effort particulier en matière d'interactivité qui s'est traduit par :

- l'absence délibérée de menus (voir ci-dessous les conséquences de l'absence de menus)
- un affichage dynamique commandé par une souris (à rapprocher de l'affichage élastique d'un segment)
- le défilement des différentes solutions possibles si la première proposée , qui paraissait la plus vraisemblable , n'est pas celle souhaitée par l'utilisateur
- le déplacement , avec le même affichage dynamique , d'une cotation existante

- la possibilité de forcer une cote à une valeur différente de celle calculée

- la possibilité d'afficher une cotation provisoire (demande de renseignements) .

Les cotations devaient également permettre de tester et de valider certaines facilités de dialogue du logiciel , à savoir :

a - pouvoir abandonner l'action à tout moment

b - pouvoir suspendre (et non abandonner) une quelconque action le temps d'en réaliser une autre.

EXEMPLE : cotation entre deux cercles .
- - - - - - - - - - - - - - - - - - - -
L'utilisateur

(1) montre le menu 'COTATIONS'

(2) montre un cercle existant
OU
en crée un sans pour autant abandonner l'action en cours et devoir recommencer en (1) (rendu possible par (b))

(3) montre un deuxième cercle
OU
en crée un sans pour autant abandonner l'action en cours et devoir recommencer en (1) (rendu possible par (b))

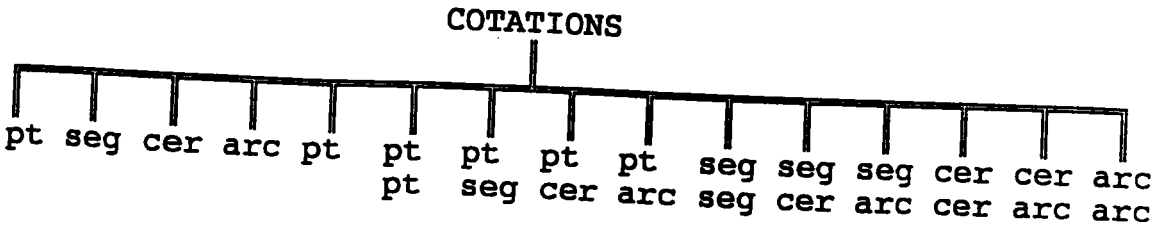
(4) voit s'afficher une cotation indiquant la distance entre les centres (entraxe) .

(5) change , en déplaçant la souris , la position de la valeur de cote et , avec elle , l'ensemble de la cotation jusqu'à ce qu'il la juge bien située (pas de chevauchement avec la scène ...)
ET , le cas échéant ,
appuie sur la barre espace pour voir défiler les autres cotations possibles , à savoir , dans notre exemple , la distance entre les points les plus éloignés des deux cercles , celle entre les deux points les plus proches , etc ...
ET / OU
tape une série de caractères qui s'ajoutent aux pré et post textes .

(6) valide son choix avec un bouton de la souris .

CONSEQUENCES DE L'ABSENCE DE MENUS

Nous avons choisi de ne pas encombrer les cotations d'un menu arborescent du type :



Nous avons en effet préféré , parce que cela nous semble d'un usage plus pratique , que le logiciel déduise automatiquement du nombre et de la nature du (des) argument(s) désigné(s) , le type de cotation souhaité .

Privé du contexte d'un sous-menu , on ne connait donc pas à l'avance le nombre d'objets à coter ni , par conséquent , le nombre d'appels nécessaires à la primitive "INTERACTION" .

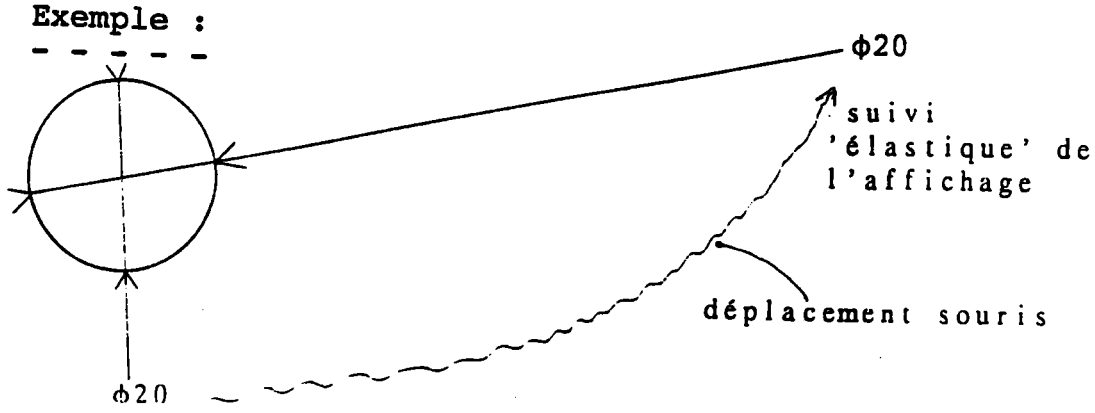
Ce problème a automatiquement trouvé sa solution dans le fait que toutes les interventions de l'utilisateur se font par l'intermédiaire d'UNE SEULE primitive retournant autant d'informations que possible (objets récupérés , coordonnées du pointé , ...) : à la deuxième interaction, il suffit donc de tester les paramètres retournés : si aucun objet n'est récupéré , l'utilisateur a sous-entendu , en montrant la position initiale de la cote , que la cotation ne mettra qu'un élément en jeu ; si par contre on recueille un objet , on en prend note et c'est le pointé qui a servi à désigner l'objet qui donne aussi la position initiale de la cote .

Il faut remarquer que ce principe , qui évite plusieurs validations à l'utilisateur sans rendre sa tâche plus contraignante , peut se généraliser à un nombre quelconque d'arguments .

AFFICHAGE DYNAMIQUE.
- - - - - - - - - - -

Par affichage dynamique , nous entendons que l'utilisateur , après qu'il ait indiqué l'endroit où doit s'afficher la valeur de cote , peut encore , notamment pour des raisons d'esthétique , déplacer l'ensemble du tracé (traits de rappel , valeur , pré et post-textes) en bougeant la souris , cette dernière restant liée au point d'affichage de la valeur de cote .

Exemple :
- - - - -



Il est donc possible de situer exactement , rapidement et d'une manière très agréable les tracés de cotation .

IV. CONCLUSION

Nous avons présenté dans cet article quelques éléments du dialogue homme-machine dans SACADO, en essayant de montrer les facilités apportées à l'utilisateur.

La réalisation d'un logiciel de dialogue et de visualisation propre au système, tient compte des expériences menées dans le domaine des normes telles que GKS ou PHIGS, mais elle s'en différencie aussi bien au niveau des primitives que de la mise en oeuvre. Elle intègre également des fonctionnalités propres à certains systèmes sur micro-ordinateurs (menus dynamiques, multi-fenêtrage...) [LEV 85] .

L'expérience que nous avons menée avec la maquette opérationnelle 2D de SACADO tend à prouver que cette approche apporte un grand confort à l'opérateur, un apprentissage aisé et des possibilités étendues par rapport au systèmes classiques .

BIBLIOGRAPHIE :
==========

[SAC 87] Y. Gardan, J.P. Jung, J.N. Kolopp, C. Minich, W. Totino
'La maquette SACADO 2D: architecture générale et dialogue.'
Rapport de recherche LRIM
Université de METZ - Février 1987 -

[GAR 86] Y. Gardan
'SACADO: Système Adaptatif de Conception Assistée et de Développement par Ordinateur : présentation générale.'
Rapport de recherche LRIM
Université de METZ - Aout 1987 -

[END 84] G. Enderle , K. Kansy , G. Pfaff
'GKS , the graphics standart'
Computer graphics programming , Springer Verlag Berlin 1984

[KRZ 86] T. Krzewina (GIXI Orsay)
'Une étude comparative de PHIGS et GKS'
Actes MICAD 1986 , pp 111-124

[TOT 86] W. Totino
'Bases d'un système de C.A.O. ouvert : application au dialogue'
Rapport de D.E.A.
Université Louis Pasteur Strasbourg
- Septembre 1986 -

[LEV 87] G. Levy
'MACINTOSH et LISA 2'
Mac Graw-Hill 1985

BIBLIOGRAPHIE

BIBLIOGRAPHIE

[AGO 76] M.K. Agoston
Algebraic topology : a first course
Marcel Dekker , INC New York and Basel

[AME 61] B. Amerein , J.C. Spehner
Un algorithme de déformation interactive utilisant le cadencement par les données
Revue de CFAO et d'infographie Vol 4 No 4 pp 59..79

[BAU 74] B.G. Baumgart
Geometric modeling for computer vision
Thèse de l'Université de Stanford 1974 Référence 75-6806

[BRA 82] I.C. Braid
Notes on a geometric modeller
CAE '82 Fevrier 1982

[BRO 85] W.F. Bronsvoort , F. Klok
Ray tracing generalized cylinders
ACM transactions on graphics Vol 4 No 4 Octobre 1985 pp 291..303

[BRU 90] P. Brunet , I. Navazo
Solid representation and operation using extended oct-trees
ACM transactions on graphics Vol 9 No 2 Avril 1990 pp 170..197

[BUC 88] B. Buchberger
Algebraic methods for geometric reasoning
Ann. Rev. Comput. Sci. Vol 3 1988 pp 85..119

[CAV 90] J.C. Cavendish , S.P. Marin
Feature-based techniques for use in the design of functional surfaces
Revue de CFAO et d'infographie Vol 5 No 2 Octobre 1990 pp 59..67

[CHO 84] B.K. Choi , M.M. Barash , D.C. Anderson
Automatic recognition of machined surfaces from a 3D solid model
Computer Aided Design Vol 16 No 2 Mars 1984 pp 377..387

[CHO 90] B.K. Choi , C.S. lee
Sweep surfaces modelling via coordinate transformations and blending
C.A.D. Vol 22 No 2 Mars 1990 pp 87..96

[CHU 90] S.H. Chuang , M.R. Henderson
Three dimensional shape pattern recognition using vertex classification and vertex-edges graphs
Computer Aided Design Vol 22 No 6 Août 1990 pp 377..387

[COQ 87a] S. Coquillart
Computing offsets of B-splines curves
C.A.D. Vol 19 Juillet 1987 pp 305..

[COQ 87b] S. Coquillart
A control-point-based sweeping technique
IEEE Computer Graphics and applications Novembre 1987 pp 36..45

[DOK 90] T. Dokken
Surface representation
Revue internationale de C.F.A.O. et d'infographie Vol 5 No 2 Hermes Paris

[DUF 88] M.R. Duffey , J.R. Dixon
Automating extrusion design : a case study in geometric and topological reasoning for mechanical design
C.A.D. Vol 20 No 10 Décembre 1988 pp 589..596

[EAS 79] C.M. Eastman , K. Weiler
Geometric modeling using Euler operators
Proceedings of first annual conference on computer graphics and CAD/CAM systems
MIT Press Cambridge Massachussets Avril 1979 pp 248..254

[ELL 89] W.S. Elliott
Computer-aided mechanical engineering : 1958 to 1988
C.A.D. Vol 21 No 5 Juin 1989 pp 275..288

[FAL 87] B. Falcidieno , F. Giannini
Extraction and organization of form features into a structured boundary model
Eurographics 87 pp 249..259

[FAR 89] R.T. Farouki , C.A. Neff
Some analytic and algebraic properties of plane offset curves
Research report RC 14364 (#64329) IBM Research Division, T.J. Watson Research Center
Yorktown Heights, N.Y. 10598

[FAU 79] I.D. Faux , M.J. Pratt
Computational Geometry for design and manufacture
Ellis Horwood Limited 1979

[FLO 89] L. De Floriani , E. Bruzzone
Building a feature-based object description from a boundary model
C.A.D. Vol 21 No 10 Décembre 1989 pp 602..610

[FOL 82] J.D. Foley, A. Van Dam
Fundamentals of interactive computer graphics
Addison Wesley 1982

[GAM 90] Y. Gardan , C. Minich
La modélisation géométrique et l'extraction de caractéristiques de formes
Actes du séminaire GAMA 90 Ecole Normale Supérieure de Cachan
Hermès (Paris) Septembre 1990

[GAR 86] Y. Gardan
La CFAO
Hermes - 1986

[GAR 88] Y. Gardan , J.P. Jung , J.N. Kolopp , C. Minich , W. Totino
Une approche nouvelle de la convivialité dans un système de C.A.O.: les principes du dialogue dans SACADO
Actes de la 7ième conférence internationale Micad Paris Hermes - 1988

[GAR 90] Y. Gardan , D. Michel , M. Sahnoune
Comparaison de formes rationnelles et non rationnelles dans la modelisation des surfaces gauches
Actes de la 9ième conférence internationale Micad Paris Hermes - 1990

[GAS 90] Y. Gardan, M. Sahnoune
L'intégration des surfaces dans les modèles de solides
Revue de CFAO et d'Infographie - Vol 5 - N. 2 - 1990

[GHO 84] P.K. Ghosh , S.P. Mudur
The brush-trajectory approach to figure specification : some algebraic solutions
ACM transactions on graphics Vol 3 No 2 Avril 1984 pp 110..134

[GOL 89] J. Goldfeather , S. Molnar , G. Turk , H. Fuchs
Near real-time CSG rendering using tree normalization and geometric pruning
IEEE Computer graphics and applications Mai 1989 pp 20..28

[GOS 88] D.C Gossard , R.P Zuffante , H. Sakurai
Representing dimensions , tolerances and features in MCAE systems
I.E.E.E Computer Graphics and Applications Mars 1988 pp 51..59

[GUI 83] L. Guibas , L. Ramshaw , J Stolfi
A kinetic framework for computational geometry
Proceedings of 24th IEEE Annual Symposium on foundations of computer science
pp 100..111 1983

[HIL 52] D. Hilbert , S. Cohn-Vossen
Geometry and the imagination
Chelsea Publishing Company New York

[HOS 85] J. Hoschek
Offset curves on the plane
C.A.D. Vol 17 1985 pp 77..

[JOS 88] S. Joshi , T.C. Chang
Graph-based heuristics for recognition of machined features from a 3D solid model
C.A.D. Vol 20 No 2 Mars 1988 pp 58..66

[JUN 90] J.P. Jung
Propagation de contraintes
Note interne; Laboratoire de Recherche en Informatique de Metz
U.F.R MIM Ile du Saulcy 57045 Metz Cedex Septembre 1990

[KAJ 83] J.T. Kajiya
New techniques for ray tracing procedurally defined objects
ACM transactions on graphics Vol 2 No 3 pp 161..181

[KLO 86] F. Klok
Two moving coordinate frames for sweeping along a 3D trajectory
Computer aided geometric design 3 1986 pp 217..229

[KOR 85] J.U. Korein
Thèse intitulée : *A geometric investigation of reach*
The MIT Press , Cambridge , Massachussets , 1985
An ACM distinguished dissertation , 1984

[LEE 83] T.W. Lee , D.C.H. Yang
On the evaluation of manipulator Workspace
Journal of mechanisms, transmissions and automation in design Vol 105 Mars 1983
Transactions of the ASME pp 70..77

[LEE 87] Y.C. Lee , K.S. Fu
Machine understanding of C.S.G. : extraction and unification of manufacturing features
I.E.E.E Computer Graphics and Applications
Janvier 1987 pp 20..32

[LOS 74] D.L. Lossing , A.L. Eshleman
planning a common data base for engineering and manufacturing
SHARE XLIII , Chicago , Août 1974

[MAN 82] M. Mantyla , R. Sulonen
GWB: A solid modeler with Euler operators
IEEE Computer graphics and applications Vol 2 No 7 Sept 1982 pp 17..31

[MAN 84] M. Mantyla
A note on the modeling space of Euler Operators
Computer vision, graphics and image processing Vol 26 No 1 Avril 1984 pp 45..60

[MAN 88] M. Mantyla
An introduction to solid modeling
Computer Science Press , Rockville , M.d. 1988

[MAR 90] R.R. Martin , P.C. Stephenson
Sweeping of three dimensional objects
C.A.D. Vol 22 No 4 Mai 1990 pp 223..234

[MAS 90] R. Maranzana , R. Soenen , C. Perron
Une application d'analyse de fabrication exploitant un modèle géométrique et technologique .
Actes de la 9ème conférence internationale
MICAD 90 pp 332..341

[MIN 88] C. Minich
Opérateurs d'extrusion et de révolution : rôle des opérateurs d'Euler
Note interne Laboratoire de Recherche en Informatique de Metz
U.F.R MIM Ile du Saulcy 57045 Metz Cedex Septembre 1990

[MON 90] C. Mony , J.C. Bocquet , P. Gosset
Evolution des solutions d'intégration technique : vers le feature modeling
Colloque international CIM 90 pp 477..486 Bordeaux France

[MOR 85] M.E. Mortenson
Geometric modeling
John Wiley and sons 1985

[NEW 79] W.M. Newman , R.F. Sproull
Principles of interactive computer graphics
Mac Graw Hill 1979

[NUT 88] A.W. Nutbourne , R.R. Martin
Differential geometry applied to curve and surface design 1. Foundations
Wiley, New York

[PEN 83] A. De Pennington , S. bloor , M. Balila
Geometric modelling : a contribution towards intelligent robots
Proceedings of 13th International symposium on industrial robots
Chicago , IL , USA 1983 pp 7.35..7.54

[PER 90] D.B. Perng , Z. Chen , R.K. Li
Automatic 3D machining feature extraction fraom 3D CSG solid input
C.A.D. Vol 22 No 5 Juin 90 pp 285..295

[PIE 87] L. Piegl
Infinite control points - A method for representing surfaces of revolution using boundary data
IEEE Computer graphics and applications Mars 1987 pp 45..55

[PUC 87] S. Pucci
Opérateurs d'Euler de construction de solides
Note interne; Laboratoire de Recherche en Informatique de Metz
U.F.R MIM Ile du Saulcy 57045 Metz Cedex ???? 1987

[RED 89] P. Redont
Representation of developable surfaces
C.A.D. Vol 21 No 1 Janvier/Février 1989 pp 13..20

[REQ 86] A.A. Requicha , S.C. Chan
Representation of geometric features , tolerancing and attributes in solid modelers based on constructive solid geometry
I.E.E.E. J. of Robotics and Automation Septembre 86 pp 156..166

[SAH 90] M. Sahnoune
Contribution à l'intégration des surfaces gauches dans les modèles de solides
Thèse de l'Université de Metz 20 Décembre 1990

[SAM 89] K. Sambandan , K. Kedem , K.K. Wang
Generalized planar sweeping of polygons
Technical report TR 89-1062 Department of computer science
Cornell University Ithaca New York

[SEI 88] H.P. Seidel
Automatic detection of closed parametric surfaces without interior
Actes du congrès Eurographics '88 1988 pp 93..103
Elsiviers science publishers B.V. (North-Holland)

[SEO 90] Y. Seok Suh , Kunwoo Lee
NC Milling tool path generation for arbitrary pockets defined by sculptured surfaces
C.A.D. Vol 22 No 5 Juin 1990 pp 273..284

[SHI 82] Y. Shiroma , N. Okino , Y. Kakazu
Research on 3-D geometric modeling by sweep primitives
Actes du congrès CAD 82 Butterworth Brighton U.K. 1892

[STR 90] I. Stroud
Modelling with degenerate objects
C.A.D. Vol 22 No 6 Juillet/Aout 1990 pp 344..351

[TIP 78] TIPS Working Group (N. Okino , Y. Kakazu, H. Kubo et a.)
Technical Information Processing System : TIPS-1
Hokkaido university , Institute of precision engineering
Sapporo. 060 Japan Mars 1978

[TOT 89] W. Totino
Contribution à la modélisation de l'interface homme-machine dans un système de C.A.O.
Thèse de l'Université de Metz 6 Octobre 1989

[VAN 82] C. Van Wik
Clipping to the boundary of a circular arc polygon
Disponible au Laboratoire de Recherche en Informatique de Metz
U.F.R MIM Ile du Saulcy 57045 Metz Cedex

[VAN 84] J.J. Van Wijk
Ray tracing objects defined by sweeping planar cubic splines
ACM Transactions on graphics Vol 3 No 3 Juillet 1984 pp 223..237

[VAN 85] J.J. Van Wijk
Ray tracing objects defined by sweeping a sphere
Computer and graphics Vol 9 No 3 1985 pp 283..290

[VER 87] A. Verrount
Visualization algorithm for C.S.G. polyhedral solids
C.A.D. Vol 19 1987 pp 527..

[VOG 89] M. Vogel
Modelling of prismatic parts in an integrated CAD/CAM system
Actes de la 8ième conférence internationale Micad Paris Hermès-1989

[VOS 85] D.L. Vossler
Sweep-to-CSG conversion using patterns recognition techniques
IEEE Computer graphics and applications Août 1985 pp 61..68

[WAN 86] W.P. Wang , K.K. Wang
Geometric modeling for swept volume of moving solids
I.E.E.E. Computer Graphics and Applications Décembre 1986 pp 8..16

[WEL 87] J.D. Weld
Geometric investigation of swept volumes with application to polyhedral objects
PhD Thesis , Cornell University , Ithaca , New York August 1987

[WIL 85] P.R. Wilson
Euler formulas and geometric modeling
I.E.E.E. Computer Graphics and Applications Août 1985 pp 24..36

[WIL 89] H. Wilson , K. Deb
Inertial properties of tapered cylinders and partial volumes of revolution
C.A.D. Vol 21 No 7 Septembre 1989 pp 456..462

[WOO 88a] J.R. Woodwark
Eliminating redundant primitives from set theoretic solid models by a consideration of constituents
I.E.E.E. Computer Graphics and Applications Mai 1988 pp 38..47

[WOO 88b] J.R. Woodwark
Some speculations on feature recognition
C.A.D. Vol 20 No 4 Mai 1988 pp 189..196

RESUME

La description de formes est une fonctionalité fondamentale d'un système de C.F.A.O. et est souvent illustrée par l'approche ensembliste, consistant à décrire un objet par une combinaison d'autres volumes dits primitifs.

D'autres opérateurs ont pourtant, théoriquement au moins, une puissance de description toute aussi considérable.

L'objectif de ce mémoire est de mettre en évidence les possibilités théoriques et pratiques de l'un de ces descripteurs de formes, **l'opérateur d'extrusion** (ou balayage de l'espace).

A cet effet, on donne dans un premier chapitre une définition de l'extrusion, qui consiste à décrire une forme à l'aide d'un objet, dit **générateur**, qui se déplace en se déformant, engendrant ainsi une ligne, une surface ou un volume.

On discute ensuite des moyens de stocker un objet ainsi défini dans un modèle géométrique de type arbre de construction ou par les limites, ce qui permet, dans une troisième partie, de détailler les algorithmes effectivement implantés.

Les trois premiers chapitres considèrent donc l'extrusion comme un opérateur d'initialisation du modèle et purement dédié à la conception.

Le dernier chapitre prend le contre-pied des précédents et montre que dans l'étape de préparation à la fabrication, la démarche est inverse puisque l'on calcule, en s'aidant du modèle géométrique issu de la conception, les parcours d'outils qui usinent la pièce, parcours qui correspondent à autant d'extrusions.

Cette incursion dans le domaine de la fabrication est le prétexte d'une réflexion sur l'intégration de la production et aboutit à la nécessité d'un modèle **produit**.

Résumé:

La description de formes est une fonctionalité fondamentale d'un système de C.F.A.O. et est souvent illustrée par l'approche ensembliste, consistant à décrire un objet par une combinaison de volumes dits primitifs.

D'autres opérateurs ont pourtant, théoriquement au moins, une puissance de description toute aussi considérable.

L'objectif de ce mémoire est de mettre en évidence les possibilités théoriques et pratiques de l'un de ces descripteurs de formes, ***l'opérateur d'extrusion*** *(ou balayage de l'espace).*

A cet effet, on donne dans un premier chapitre une définition de l'extrusion, qui consiste à décrire une forme à l'aide d'un objet, dit ***générateur****, qui se déplace en se déformant, engendrant ainsi une ligne, une surface ou un volume.*

On discute ensuite des moyens de stocker un objet ainsi défini dans un modèle géométrique de type arbre de construction ou par les limites, ce qui permet, dans une troisième partie, de détailler les algorithmes effectivement implantés.

Les trois premiers chapitres considèrent donc l'extrusion comme un opérateur d'initialisation du modèle et purement dédié à la conception.

Le dernier chapitre prend le contre-pied des précédents et montre que dans l'étape de préparation à la fabrication, la démarche est inverse puisque l'on calcule, en s'aidant du modèle géométrique issu de la conception, les parcours d'outils qui usinent la pièce, parcours qui correspondent à autant d'extrusions.

Cette incursion dans le domaine de la fabrication est le prétexte d'une réflexion sur l'intégration de la production et aboutit à la nécessité d'un modèle ***produit****.*

Mots clés:

C.A.O. (Conception Assistée par ordinateur), extrusion (sweeping), extrusion généralisée, balayage de l'espace.
Opérateurs d'Euler.
Prismes généralisés, volumes de révolution.
Extraction de caractéristiques de forme, modélisation produit.