Opérations booléennes sur polyèdres : évaluation d'arbres CGS

Didier Badouel, Gérard Hégron

INRIA

UNITÉ DE RECHERCHE
INRIA-RENNES

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tél.:(1) 39 63 55 11

Rapports de Recherche

N° 839

OPERATIONS BOOLEENNES SUR LES POLYEDRES : EVALUATION D'ARBRES CSG

Didier BADOUEL
Gérard HEGRON

AVRIL 1988

RR 839

IRISA

INSTITUT DE RECHERCHE EN INFORMATIQUE
ET SYSTÈMES ALÉATOIRES

Campus Universitaire de Beaulieu
35042 - RENNES CÉDEX
FRANCE
Téléphone : 99 36 20 00
Télex : UNIRISA 950 473 F
Télécopie : 99 38 38 32

Publication Interne n° 404
Avril 1988
46 Pages

Opérations booléennes sur les polyèdres : évaluation d'arbres CSG

Didier BADOUEL Gérard HEGRON

18 avril 1988

Résumé

L'évaluation des opérations booléennes sur les polyèdres constitue un rouage important des Systèmes de modélisation géométrique (GMS). Cette évaluation utilise une description de solide dans un modèle de type CSG (Constructive Solid Geometry), et fournit comme résultat une représentation polyèdrique constituant une structure de données performante dans le domaine de la visualisation. La complexité des algorithmes de calcul des opérations booléennes est quadratique si aucune subdivision spatiale n'est utilisée. Dans ce rapport, nous présentons une nouvelle méthode d'évaluation utilisant une structure de subdivision spatiale que nous appelons *Octree Booléen*. Cette structure a pour but de limiter l'évaluation des opérations booléennes dans un 'espace minimum de calcul' dans lequel les frontières des primitives se coupent effectivement et dont l'évaluation est nécessaire à la construction de l'objet final. La dernière partie du rapport concerne les calculs effectifs des opérations booléennes. Du fait de l'aspect local de l'évaluation, les problèmes de cohérence de données entre les différents sous_espaces doivent être éludés. En premier lieu, nous allons présenter les outils de modélisation constituant le contexte de l'étude, ainsi que les différentes méthodes utilisées pour l'évaluation des opérations booléennes.

Set Operations on Polyhedra : Evaluation of CSG Trees

Abstract

Set operation evaluation on polyhedral objects is an important component of Geometric Modeling System (GMS). CSG (Constructive Solid Geometry) modeling can be used for a solid description and the result will be a polyhedral representation which provides an efficient data structure in display field. With no use of spatial coherency, computational complexity of set operation is quadratic. In this paper, we are introducing a new space subdivision scheme called *Boolean Octree* which performs set operations on polyhedra with a linear complexity. This structure aims at limiting set operation evaluation in a 'minimal space of calculation' where primitive boundaries intersect each other and where resulting evaluation participates in the construction of the final resulting object. The last part of this report concerns the effective set operation evaluation. The local nature of this

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE
(L. A. 227)
UNIVERSITÉ DE RENNES 1 I. N. S. A. DE RENNES

INSTITUT NATIONAL DE RECHERCHE
EN INFORMATIQUE ET EN AUTOMATIQUE
(LABORATOIRE DE RENNES)

our study is the effective set operation evaluation. The local nature of this evaluation involves data coherency upholding between the different local spaces. First, we introduce our study with a presentation of modeling's tools, and have a look at previous set operation evaluation methodologies.

Table des matières

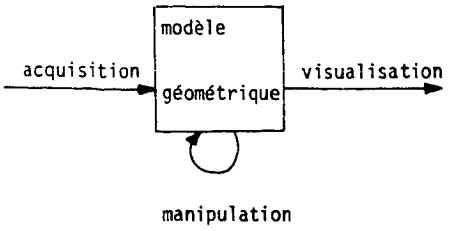


Figure 1 : Représentation schématique d'un GMS.

1 Modélisation

La création et l'utilisation de scènes tridimensionnelles se sont développées au cours de ces dernières années. Les domaines d'application classiques sont la conception assistée par ordinateur (CAO) et la fabrication assistée par ordinateur (FAO). La modélisation géométrique regroupe l'ensemble des méthodes qui permettent la représentation interne et la gestion des scènes tridimensionnelles. La diversité des domaines d'application a motivé l'étude de ces systèmes:

- étude d'un schéma générique de système de modélisation.
- étude de modèles et de procédures répondant à des besoins spécifiques.

La performance de ces systèmes correspond à différents critères d'efficacité dans des applications spécialisées (visualisation temps réel, calcul de propriétés ...) et dans la richesse d'utilisation (diversité des traitements).

1.1 Système de modélisation géométrique de solides

Un GMS (Geometric Modeling System) possède quatre composantes principales (Fig. 1) :

- une structure symbolique (modèle) qui est une représentation géométrique des solides.
- des utilitaires de création d'objets, de scènes.
- des procédures de visualisation.
- des procédures de calcul, d'analyse des solides. Par exemple, calcul des propriétés physiques d'un solide (volume, centre d'inertie ...).

Au coeur de ce système se trouve le modèle géométrique. La performance des procédures d'analyse est conditionnée par le choix de la structure symbolique. Les modèles géométriques représentent de façon abstraite des sous_ensembles de l'espace euclidien (E^3). Très peu de sous_ensembles de E^3 représentent des solides physiques. A.A.G. Requicha donne dans [Req80] une liste des propriétés des solides, propriétés que doivent respecter les modèles géométriques :

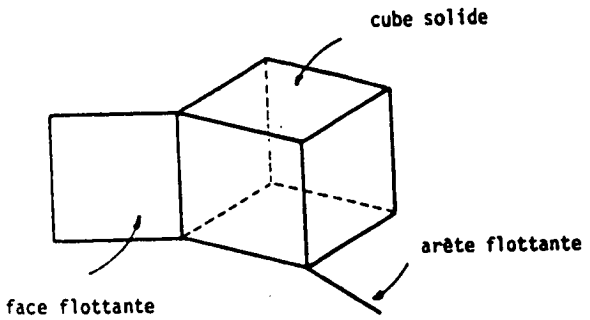


Figure 2 : Exemple d'objet non solide possédant des éléments *pendants*.

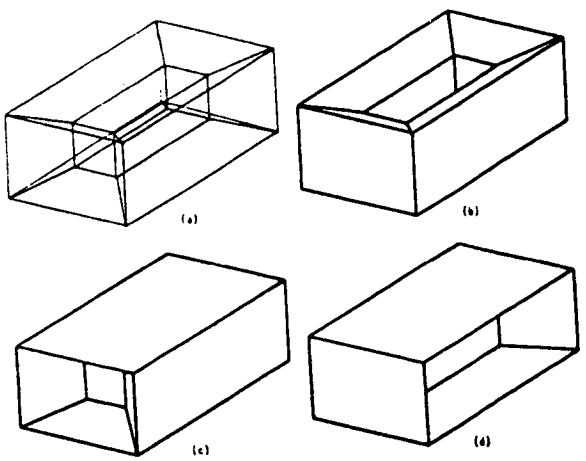


Figure 3 : Ambiguïté du modèle fil_de_fer.

1. la *rigidité*: un solide doit avoir une forme invariante, indépendante de la position ou de l'orientation.

2. l' *homogénéité tridimensionelle*: un solide ne doit pas posséder d'éléments '*pendants*'('*dangling*'). L'exemple de la figure 2 ne représente pas un solide.

3. la *finitude*: un solide doit occuper une partie finie de l'espace.

4. la *clôture*: maintien de la clôture (topologique) par rapport aux opérations de positionnement et aux opérations booléennes.

5. la *finitude descriptive*: un solide doit être décrit à l'aide d'un nombre fini d'éléments.

6. le *déterminisme des frontières*: la frontière d'un solide doit déterminer sans ambiguité ce qui est interne au solide. Le modèle 'fil_de_fer', où un objet est représenté par des

segments de droite, est un modèle ambigu. Dans l'exemple (cf Fig.3), quel solide représente l'objet (a) ? est-ce le solide (b), (c) ou (d) ?

1.2 Modèles

Parmi les modèles non ambigus et valides (permettant le respect des propriétés des solides), nous allons présenter les trois familles que nous utiliserons dans notre rapport :

- représentation par arbre CSG, qui sera le modèle initial des objets manipulés.
- représentation par frontière, qui sera la forme finale des objets traités.
- représentation par arbre octal dont la méthodologie sera exploitée pour le passage de la représentation initiale à la représentation finale des objets.

1.2.1 Représentation par arbre CSG ('Constructive Solid Geometry')

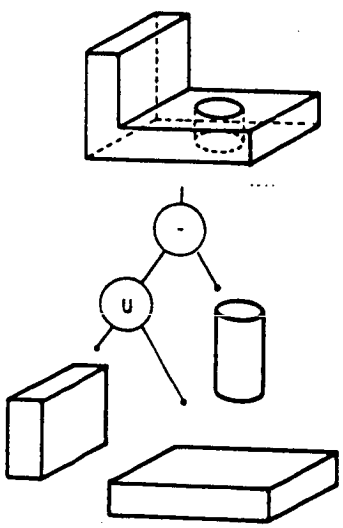


Figure 4 : Exemple de solide représenté par un arbre CSG.

Un solide est représenté dans le modèle CSG par une expression booléenne de primitives géométriques simples. Ces primitives peuvent être, par exemple, le parallélépipède, la sphère, le cylindre et le cône. En plus des dimensions de la primitive, les feuilles de l'arbre contiennent les informations concernant son positionnement dans la scène. Ce modèle est non ambigu. De plus, cette représentation est valide si les opérations sont *régularisées* : une opération régulière fournit comme résultat un objet n'ayant aucun élément *'pendant'*(cf 2.1.1). Ce modèle est bien adapté à la construction interactive de solides. Cependant, la visualisation et l'analyse nécessitent des procédures du type *Lancer_de_rayon* (*Ray casting*) coûteuses en temps de calcul.

1.2.2 Représentation par frontière

Dans ce modèle, un solide est représenté par l'ensemble frontière entre sa partie interne et sa partie externe. La frontière est constituée d'un ensemble de *faces*, reliées par des *arêtes (edges)*. Les extrémités des arêtes sont appelées *sommets (vertices)*.

Pour que le modèle soit valide, il doit y avoir concordance entre les éléments de la représentation et la topologie du solide. Pour celà, les règles suivantes doivent être vérifiées:

1. chaque face doit être un sous_ensemble de la frontière (topologique) du solide.
2. la réunion de toutes les faces doit constituer l'ensemble de la frontière.
3. chaque face doit être un sous_ensemble d'une primitive surfacique S_i. Sur ce point, A.A.G. Reqicha montre dans [RT78] que tout modèle de représentation par frontière possède un nombre fini de primitives surfaciques.
4. la représentation ne doit posséder ni face '*pendante*', ni arête '*pendante*', ni point isolé.

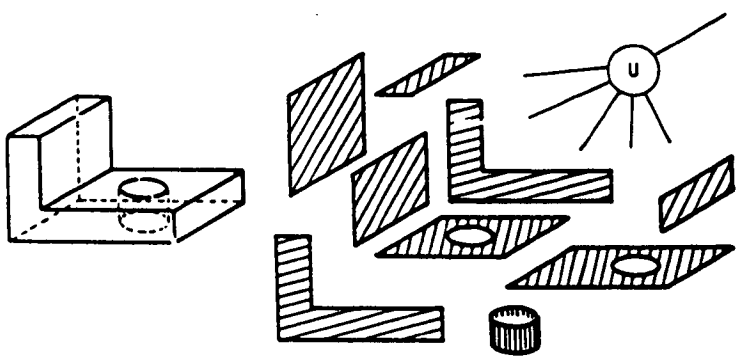


Figure 5 : Exemple de solide représenté à l'aide de facettes.

De nombreuses applications utilisent le modèle Frontière où les faces sont des faces polygonales planes: c'est la modélisation *polyèdrique*.
Dans ce cas, la validité consiste à respecter les points suivants:

- pas d'arête ni de face pendante: cette condition est vérifiée si chaque arête unit deux faces et deux sommets et si chaque face possède autant d'arêtes que de sommets.
- pas de faces qui se coupent.
- les faces doivent être orientables. Le plan support d'une face détermine deux demi_espaces, l'un orienté vers l'intérieur de l'objet, l'autre vers l'extérieur. Le modèle doit contenir l'information nécessaire à déterminer l'orientation des faces.

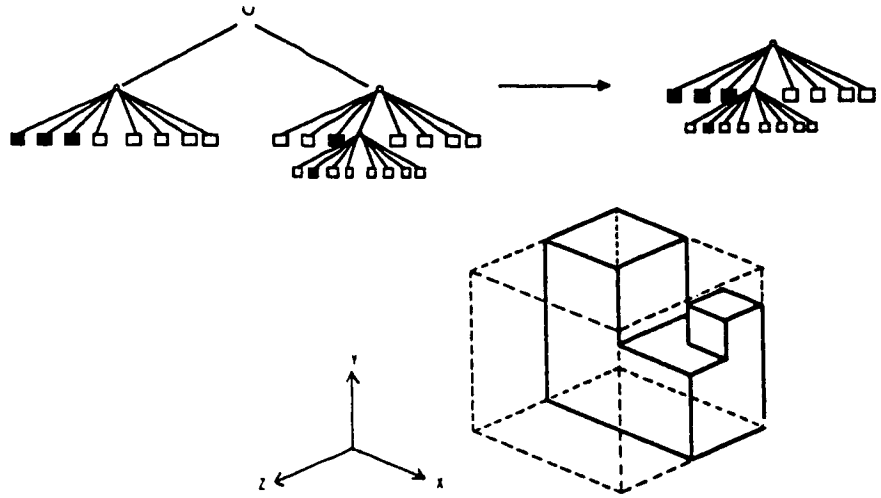


Figure 6 : Exemple d'opération booléenne sur des arbres octaux.

La représentation polyèdrique est la plus adaptée à l'heure actuelle pour la visualisation rapide des solides. L'inconvénient de cette méthode est son manque de convivialité: il n'est pas aisé de décrire un solide comme une énumération de polygones.

1.2.3 Représentation par arbre octal (OCTREE)

Un solide est représenté dans le modèle Octree par un ensemble de cubes résultant de la subdivision récursive en octants du cube englobant l'objet. Si un cube est partiellement à l'intérieur du solide, on le subdivise en huit fils appelés *octants*. Le processus de subdivision s'arrête si le cube est situé soit à l'intérieur (octant NOIR) soit à l'extérieur (octant BLANC) de l'objet ou si la taille minimale de décomposition est atteinte (octant GRIS). Cette représentation est facilement calculable à partir d'un autre mode de représentation. Elle est intéressante pour le calcul des propriétés physiques et pour la réalisation des opérations booléennes. La figure 6 illustre un exemple d'opération booléenne effectuée sur des arbres octaux.

1.2.4 Discussion autour des modèles

Les avantages principaux des modèles présentés ci-dessus sont les suivants:

- CSG : création interactive.
- Frontière : visualisation rapide.
- Octree : calcul des propriétés physiques.

Les inconvénients principaux:

- CSG : temps de calcul pour la visualisation.
- Frontière : création de solides.
- Octree : approximation du solide.

Nous pouvons remarquer qu'il n'existe pas de modèle '*idéal*' répondant simultanément aux contraintes liées à la création, à la manipulation ou à la visualisation. Cependant des représentations hybrides permettent de combiner plusieurs modèles. Deux exemples peuvent être cités :

- CSG/Frontière : arbre CSG où les feuilles sont soit une primitive solide soit une représentation par Frontière d'un solide (non_primitive).
- CSG/Balayage : arbre CSG où les feuilles peuvent être des représentations par Balayage de solides.

Les modèles hybrides posent des problèmes de traitement. Du fait de la non homogénéité des représentations, les procédures devront pouvoir s'éxécuter sur les différentes structures. Pour pallier les carences de tel ou tel modèle une solution consiste à mettre en oeuvre des mécanismes de conversion d'un modèle dans un autre qui permettront de disposer de plusieurs modes de reprsentation du même objet.

1.3 Opérations booléennes

Les opérations booléennes, introduites dans la modélisation CSG, permettent la composition de solides élémentaires (primitives) pour l'élaboration de nouveaux objets plus complexes. Elles permettent également la détection d'interférence spatiale (collision...) entre objets.

Du fait :

- de la rapidité de visualisation des objets modélisés par leurs frontières;
- mais de l'attrait pour la création et la représentation d'un objet à partir d'opérateurs de construction booléens;

il apparaît intéressant de mettre en oeuvre la conversion du modèle CSG dans la représentation par frontière.

La conversion de la représentation des primitives solides de l'arbre CSG dans leur représentation par frontière peut être effectuée :

- soit lors de l'évaluation des opérations booléennes (PADL_2);
- soit avant l'évaluation des opérations booléennes (GWB,DMI).

Nous avons choisi la deuxième méthode, c'est-à-dire réaliser les opérations booléennes sur des primitives transformées préalablement dans le modèle Frontière. Une présentation plus détaillée des méthodes utilisées dans différents GMS (PADL_2, GWB, DMI), ainsi que la justification du choix effectué, est faite dans la section 2.2.

Les représentations choisies pour cette étude sont les suivantes :

- utilisation d'un modèle CSG décrit à l'aide du langage LGRC développé à l'IRISA (cf [BAP86]).
- approximation polyèdrique des primitives solides (cube, sphère, etc.)pour le traitement des opérations booléennes.

Les points importants pour l'étude et la réalisation des opérations booléennes sont les suivants :

- le traitement doit s'effectuer sur un modèle le plus 'complet' possible : ceci conditionne la richesse des informations contenues dans les structures de données représentant les solides (nous y reviendrons dans les sections 3.2 et 4).
- le résultat de l'opération doit respecter les propriétés des solides : une opération booléenne sur deux solides *valides* doit donner comme résultat un nouveau solide *valide*. Dans la section suivante, nous allons présenter comment assurer cette validité à l'aide des opérations régulières.

2 Opérations booléennes de polyèdres

2.1 Définitions

opérateurs ensemblistes	opérateurs booléens
A ∪ B	a ou b
A ∩ B	a et b
A − B	a et ¬b

Les opérations booléennes sur les solides sont aparentées aux opérateurs ensemblistes (∪, ∩, −); et peuvent être présentées comme suit:

- A ∪ B = ensemble des points appartenant à l'un ou l'autre des deux solides.
- A ∩ B = ensemble des points appartenant aux deux solides.
- A − B = ensemble des points du solide A à l'extérieur du solide B.

Malheureusement, cette description n'est pas correcte. Elle produit des objets qui ne sont pas nécessairement des solides: l'exemple 7 illustre la présence d'éléments 'parasites' (arêtes pendantes, faces pendantes ou points isolés) dans le résultat d'une opération booléenne. Il est donc nécessaire de définir des opérateurs dits *réguliers* conservant la validité des solides. Ces opérations régulières sont définies à l'aide de propriétés topologiques.

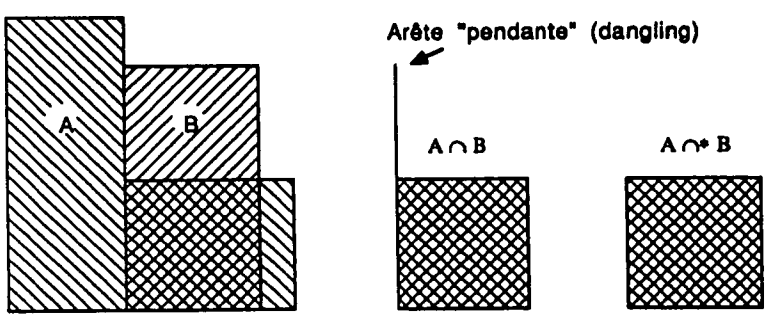


Figure 7 : Opération booléenne régularisée.

2.1.1 Notions de topologie

Le résultat d'une opération booléenne doit être un solide, c'est-à-dire représentant un sous_ensemble de E^3 topologiquement clos et régulier.
Tout d'abord, donnons quelques définitions topologiques :

- $p \in iX$: un point p est un élément *intérieur* à un ensemble X s'il existe un voisinage de p contenu dans X.

- $p \in kX$: un point p est un élément de la *fermeture* de X si tous les voisinages de p contiennent un point de X.

- $p \in bX$: un point p est un élément *frontière* de X si p est un élément de kX et de k(cX). cX est le complémentaire de X.

On rappelle qu'une partie V de E est appelée *voisinage* de p dans E s'il existe une partie ouverte U de E telle que x $\in U$ et $U \subset V$. Cela correspond, dans R^n, à une boule de rayon ϵ centrée sur p.
Une partie U est *ouverte* si, pour tout $x_0 \in U$, il existe un $\epsilon > 0$ tel que tout point x de E vérifiant $d(x_0,x) < \epsilon$ appartienne à U.

A l'aide de celà, R. B. Tilove définit les propriétés à respecter (cf [Til80]) :

- un ensemble X sera dit *régulier* si $X = kiX$;

- la régularisation d'un ensemble X est défini par $rX = kiX$.

On notera :

- $A \cup^* B = r(A \cup B) = ki(A \cup B)$

- $A \cap^* B = r(A \cap B) = ki(A \cap B)$

- $A -^* B = r(A - B) = ki(A - B)$

2.1.2 Classification

Lorsque l'on travaille avec une représentation par Frontière, on peut utiliser la propriété suivante :

- la frontière de $(A\ op\ B)$ est contenue dans la réunion de bA et de bB (ensembles des frontières de A et de B).
  Ainsi, la frontière de n'importe quelle composition est contenue dans l'union des frontières de ses primitives.
  (Cette propriété est prouvée formellement dans [RT78].)

Dans le modèle Frontière, les opérations booléennes peuvent être traitées comme un problème de classification et de sélection. Pour réaliser une opération entre deux solides, on compare chacune des frontières de l'un par rapport à l'ensemble des frontières de l'autre solide, pour déterminer qu'elle est la partie à l'intérieur, la partie à l'extérieur et enfin la partie sur la frontière de l'autre objet. On sélectionne ensuite les éléments à conserver pour constituer l'objet résultant.

Lors du traitement des opérations booléennes, différents niveaux de classification peuvent intervenir :

- interférence entre deux solides;

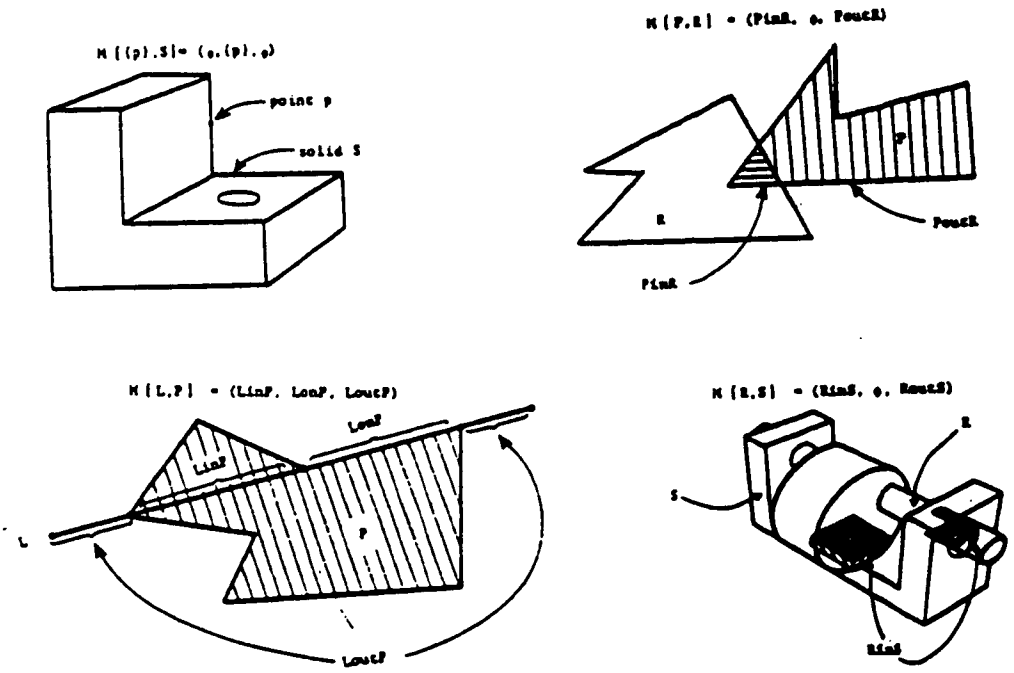


Figure 8 : Exemples de classification (tiré de [Til80]).

- intersection entre deux polygones;
- classification d'un segment de droite par rapport à un solide;
- position d'un point vis-à-vis d'un solide.

Ces différents problèmes peuvent être décrits à l'aide d'une fonction de classification, noté *M(X,S)*, opérant sur deux ensembles de points: l'ensemble candidat (*X*), et l'ensemble de référence (*S*). Le résultat de l'opération est un découpage de l'ensemble candidat en trois portions :

- *X in S* : le sous_ensemble de *X* situé à l'intérieur de *S*.
- *X out S* : le sous_ensemble de *X* situé à l'extérieur de *S*.
- *X on S* : le sous_ensemble de *X* situé sur la frontière de *S*.

Sur la figure 8, on peut voir quelques exemples de classification avec des ensembles de dimensions 1,2 ou 3.

D'une manière plus formelle :

- $X\ in\ S = X \cap^* iS = k_x i_x (X \cap i_s S)$
- $X\ out\ S = X \cap^* bS$

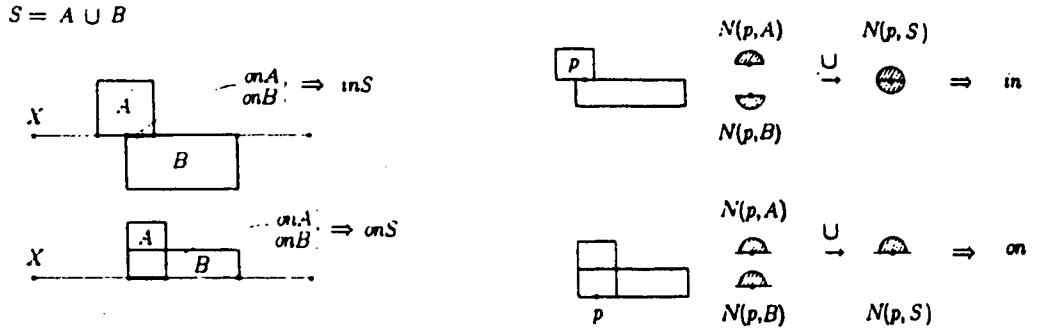


Figure 9 : Ambiguïté on/on : exemple et résolution (tiré de [Til80]).

- $X\ on\ S = X \cap^* cS$

Pour la réalisation des opérations booléennes, on souhaite :
connaissant (*XinA, XonA, XoutA*) et (*XinB, XonB, XoutB*), déterminer (*XinS, XonS, XoutS*) avec S = A< *op* >*B.
Malheureusement, lorsque les objets possèdent des frontières qui se chevauchent, il y a des problèmes d'ambiguïté (appelés *on/on* ambiguïté). Dans ce cas, les éléments sont :

- soit internes à A∪B;
- soit sur la frontière de A∪B.

Pour résoudre ces ambiguïtés, il faut, lors de la classification, étudier la configuration du voisinage des éléments frontière ($\in XonS$).

L'étude du voisinage d'un point *p* frontière (appartenant à une face d'un objet), correspond à évaluer l'intersection entre la boule (de rayon ε centrée sur *p*) représentant l'ensemble des points voisins de *p* et l'ensemble des points internes à l'objet (*iX*).
L'information concernant le voisinage d'un point *p* peut être représentée par une fraction de boule. Pour résoudre les ambiguïtés *on/on*, on réalise les opérations booléennes sur ces fractions de boule (cf Fig.9) : si le résultat est une boule pleine, le point est interne au nouvel objet, si la boule est vide le point n'appartient pas au nouvel objet sinon il est sur sa frontière.

Dans la modélisation polyèdrique, la partie interne du voisinage d'un point d'une facette est une demi_sphère correspondant au demi_espace (délimité par le plan support de la facette) interne à l'objet : il suffit d'avoir la normale au plan support de la facette pour indiquer le demi_espace concerné. Classiquement, on oriente les sommets d'une facette de manière à ce que la normale pointe vers l'extérieur du solide.

Lorsque deux faces se chevauchent on distingue deux types de configuration (cf Fig. 10) :

- IDEM : type des facettes qui ont la même orientation dans les deux solides.
- OPPOSÉ : type des facettes qui ont une orientation opposée.

2.2 Méthodes existantes

Diverses méthodes de résolution des opérations booléennes existent dans les systèmes de modélisation. Le but commun est un éclatement des faces des solides opérandes pour l'élaboration des faces du solide à évaluer. Nous allons distinguer les méthodes suivant le type de classification utilisée :

- classification de type ARETE − FACE :
  c'est la procédure utilisée dans le système GWB (*Geometric Work-Bench*), développé à l'université d'Helsinki ([MS82]). Toutes les arêtes d'un solide sont classifiées vis-à-vis de l'autre solide. On obtient une série de polygones tronqués définis par les points d'intersection (arête − polygone). Puis, les faces d'un objet A (resp. B) sont classifiées par rapport à l'objet B (resp. A) : on obtient deux ensembles pour chaque solide $AinB$ et $AoutB$, les faces frontière ne sont pas retenues.
  Le processus de sélection est effectué suivant l'opération :

  - $A \cup B$: $AoutB$ et $BoutA$
  - $A - B$: $AoutB$ et $BinA$
  - $A \cap B$: $AinB$ et $BinA$

  La fermeture de l'objet est obtenue à l'aide des opérateurs d'Euler ([MS82]), et les faces coplanaires contiguës sont rassemblées en une seule. La nécessité de créer des arêtes (pour la fermeture de l'objet) tout en respectant la formule d'Euler entraine des restrictions sur le nombre de parties solides disjointes (isolées) dans le résultat d'une opération booléenne.

  Remarque : la formule d'Euler exprime la validité d'un solide :

$$v - e + f - h = 2(b - r).$$

  Avec
  v : le nombre de sommets (vertex).
  e : le nombre d'arêtes (edge).
  f : le nombre de faces.
  h : le nombre de trous sur les faces (hole).
  b : le nombre de parties disjointes de l'objet.
  r : le nombre de trous à travers l'objet.

- classification de type ARETE – SOLIDE :
  c'est la méthode utilisée pour la conversion du modèle CSG au modèle Frontière mise_en_oeuvre dans le système PADL_2. Ce système est développé à l'université de Rochester ([RV85]).
  La procédure consiste à faire les traitements suivants :

  - pour chaque face d'un solide A (resp. B), classifier chaque arête ou portion d'arête vis-à-vis du solide B (resp. A). i.e. trouver les ensembles *EinB*, *EonB* et *EoutB*.
    A partir de ces ensembles et de l'information consernant le voisinage de l'arête (cf 2.1.2), les différents éléments sont selectionnés pour constituer les arêtes du solide S ($S = A\ op\ B$).
  - pour chaque face d'un solide A (resp. B), trouver l'intersection avec les faces du solide B (resp. A). Les nouvelles arêtes créées sont également classifiées vis-à-vis du solide B (resp. A).

  La sélection des éléments est effectuée, évidemment, en fonction de l'opérateur booléen. Le problème est l'information concernant le voisinage (associée à chaque face et à chaque arête), qui doit être recalculée après chaque opération. C'est une procédure relativement complexe.

- classification de type FACE – FACE :
  c'est la méthode utilisée par D. Ayala dans le système DMI ([ABJN85]) et par D.H. Laidlaw ([LTH86]).
  Dans la méthode de D.H. Laidlaw, le traitement s'effectue sur des solides représentés à l'aide de facettes planes (polygones). Les intersections entre les polygones du solide A et les polygones du solide B sont évaluées. En fonction du segment intersection, ces facettes sont découpées, puis les facettes résultantes sont classifiées vis-à-vis de l'autre solide.

  On obtient les huit ensembles de facettes suivants :

  - *AinB, AonB (IDEM), AonB (OPPOSÉ)* et *AoutB*
  - *BinA, BonA (IDEM), BonA (OPPOSÉ)* et *BoutA*

  Le solide résultat de l'opération est un ensemble de facettes inclu dans la réunion de ces huit ensembles. Suivant l'opération, on effectue la sélection des ensembles (cf 10).

  - $A \cup B$: *AoutB, AonB (IDEM)* et *BoutA*
  - $A \cup B$: *AoutB, AonB (OPPOSÉ)* et *BinA*
  - $A \cap B$: *AinB, AonB (IDEM)* et *BinA*

  De plus, dans l'algorithme de D.H. Laidlaw la convexité des polygones, nécessaire pour les solides opérandes, est conservée lors de l'éclatement des polygones.

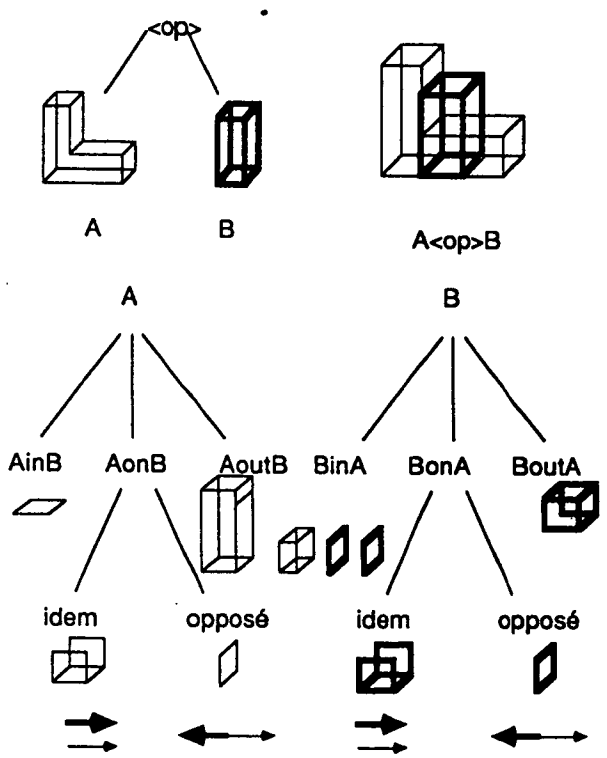


Figure 10 : Exemple de classification de solides (modélisation polyèdrique).

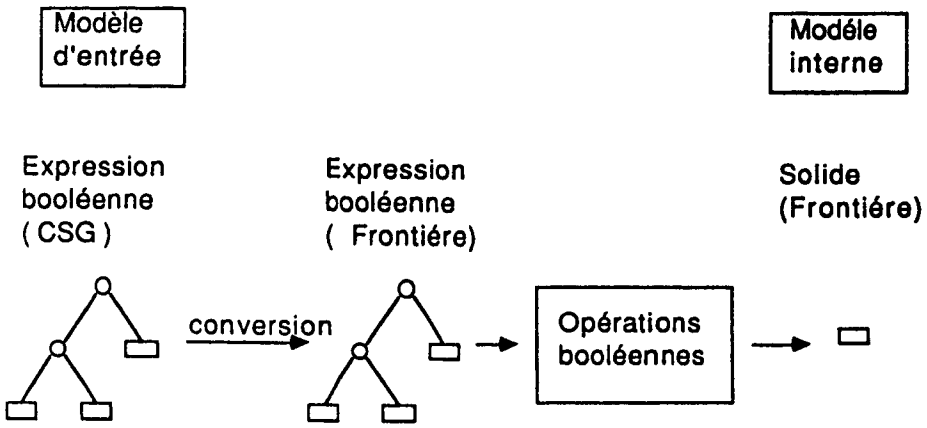


Figure 11 : Schéma général du calcul des opérations booléennes.

Choix réalisé : les avantages principaux des types de classification utilisés dans les différents GMS sont les suivants :

- GWB : classification simplifiée, pas de prise en compte des arêtes et faces frontière au niveau de la sélection des éléments frontière.
- PADL_2 : la conversion du modèle CSG au modèle Frontière est réalisée par le traitement des opérations booléennes; pas de conversion préalable des primitives dans le modèle Frontière.
- DMI : étude du voisinage pour les facettes frontière relativement simple.

Les inconvénients principaux :

- GWB : utilisation des opérateurs d'Euler pour la création des arêtes indispensables à la fermeture du solide évalué.
- PADL_2 : gestion du voisinage complexe.
- DMI : conversion des primitives dans leur représentation par frontiŕe nécessaire avant la résolution des opérations booléennes (vrai également pour GWB).

Nous avons choisi la classification de type FACE↔FACE (Ayala, Laidlaw), car elle facilite la reconstruction de l'objet. On n'a pas besoin d'utiliser les opérateurs d'Euler pour assurer la validité de l'assemblage des polygones.
Cette méthode illustre bien la propriété énoncée dans 2.1.2: l'élaboration de la frontière du nouveau solide est faite en sélectionnant des portions de la frontière des primitives.

Le schéma général de traitement est le suivant (cf 11):

- conversion de chaque primitive solide (de l'arbre CSG) dans une représentation polyèdrique (approximation polyèdrique).

- application de la procédure d'évaluation des opérations booléennes; le résultat est un polyèdre.

On rappelle le principe général de l'algorithme de D.H. Laidlaw :

- éclater les polygones de A et de B en polygones convexes qui ne se coupent pas.
- classification des polygones A $\leftrightarrow$ solide B.
- classification des polygones B $\leftrightarrow$ solide A.
- sélection des polygones suivant l'opération.

Le problème des algorithmes de classification est que chaque face (ou chaque arête) d'un solide doit être comparé à l'ensemble des faces de l'autre objet.
Pour chaque opération booléenne, on obtient une complexité de l'ordre de (M * N) (N et M sont les nombres de facettes respectifs des solides A et B). Cette complexité fait des opérations booléennes un traitement lent dès que le nombre total de facettes devient important.

Pour résoudre ce problème, D.H. Laidlaw utilise des structures de boîtes englobantes. Nous allons voir dans la section suivante que cette structure de localisation n'est pas très satisfaisante. En revanche, l'utilisation d'une subdivision spatiale permet de diminuer l'ordre de complexité de ces algorithmes.
L'étude de la subdivision spatiale (section 3) a deux buts principaux :

- ne pas effectuer les calculs de classification qui ne sont pas nécessaires à l'élaboration du solide final;
- réaliser la classification dans un espace où le nombre de faces est limité. Les calculs de classification seront alors effectués localement : le nombre de comparaisons sera fonction du nombre de faces dans ces espaces locaux.

Dans la section suivante, nous allons décrire différents modes de localisation spatiale. Parmi ceux_ci, l'étude des *Octrees exacts* (modèle géométrique d'objet) nous a permis de définir l'*Octree Booléen* qui est un processus de subdivision spatiale utilisé lors de l'évaluation de l'arbre CSG.
La description des *Octrees Booléens* est faite dans la section 3.2.

3 Subdivision spatiale

La réalisation des opérations booléennes sur des polyèdres nécessite dans le cas général des calculs d'une complexité quadratique. Pour effectuer une opération booléenne entre deux objets il faut réaliser la classification des frontières de chaque objet.

Du fait de l'aspect systématique des comparaisons dans les algorithmes de classification, l'élimination des comparaisons inutiles permet une réduction importante du nombre de calculs. La localisation d'éléments dans des espaces disjoints permet d'éviter la classification effective de ces deux éléments . Deux critères pour le choix du type de localisation spatiale peuvent être suggérés:

- un critère de simplicité: la comparaison de deux espaces doit être beaucoup plus simple que la comparaison directe des deux éléments.
- un critère d'efficacité: les espaces de localisation doivent permettre d'éliminer le maximum de comparaisons inutiles. Pour cela le type de localisation doit fournir une bonne approximation de l'espace réel occupé par un élément.

3.1 Méthodes existantes

3.1.1 Les volumes englobants

Les volumes englobants sont une approximation du volume occupé par un élément, mais ne constituent pas une subdivision spatiale. Pour celà il faudrait établir une hiérarchie entre ces volumes.
Dans les algorithmes de classification facette/solide, la comparaison avec l'ensemble des faces du solide est nécessaire dès lors que la facette possède un volume englobant intersectant soit le volume englobant du solide soit un volume englobant d'une facette du solide. La subdivision spatiale, elle, doit permettre une description des solides dans des espaces de taille inférieure à celle de la scène (avec un nombre limité de facettes). La classification pourra alors s'effectuer par rapport à la représentation locale des solides (nous reviendrons en détail sur ces notions dans la section 4).
Avec l'utilisation des volumes englobants, l'ordre de complexité des calculs de comparaison reste quadratique. Celui ci ne pourra être linéaire qu'avec l'utilisation d'une subdivision spatiale .

3.1.2 La décomposition en arbres octaux

Un Octree (cf 1.2.3) est une structure de données hiérarchique qui représente l'occupation spatiale d'un objet. Les noeuds terminaux (feuilles) de l'arbre octal sont de trois types:

- octant vide (noté BLANC): il est à l'extérieur du solide;
- octant plein (noté NOIR): il est contenu dans le solide;
- octant indéfini (noté GRIS): il est partiellement situé à l'intérieur du solide.

Pour obtenir une représentation exacte d'un objet, le processus de décomposition devrait se poursuivre indéfiniment, la seule limite étant l'arrondi fait par le calculateur; il convient donc de fixer une limite dans le niveau de décomposition qui définira le degré de résolution de la représentation.
Le problème principal, pour ce qui concerne notre étude, est la difficulté de passer d'une représentation octale à une représentation par frontière (formée ici de facettes polygonales planes). Une étude a été réalisée dans [KSY85], pour effectuer cette conversion 'représentation octale'–'représentation polyèdrique'. Le principe du traitement est d'associer aux octants noirs huits faces carrées, représentant les frontières de l'octant, puis d'assembler les faces contiguëes et de supprimer les faces internes. Cette construction est réalisée en utilisant les opérateurs d'Euler (cf 2.2).
Néanmoins, cette méthode ne permet pas de conserver la représentation exacte des polyèdres : l'approximation de la conversion octale fait du polyèdre final un objet approchant le polyèdre souhaité.

3.1.3 La décomposition en Octrees Polygonaux

Les deux inconvénients principaux d'une décomposition d'un solide en Octree sont:

- la taille très importante de la mémoire nécessaire à la représentation d'un solide.
- l'inexactitude de ce modèle: approximation du volume d'un solide par des octants de taille minimale.

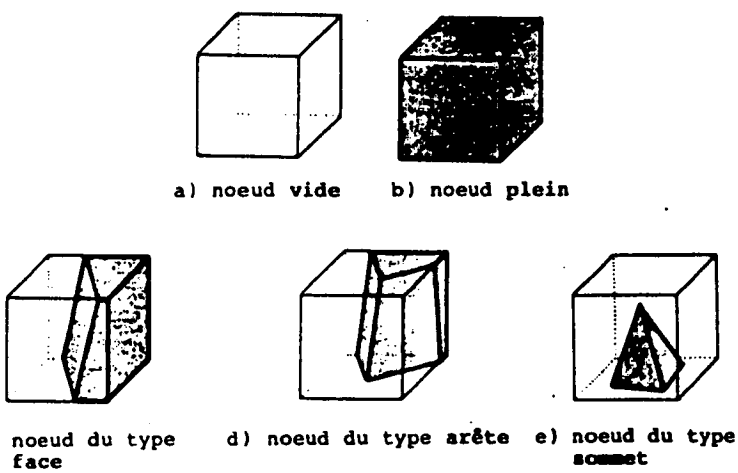


Figure 12 : Types des nœuds terminaux dans un Octree exact.

Afin de résoudre ces problèmes, il est possible d'utiliser les *Octrees Polygonaux* (ou *Octrees Exacts*). Cette décomposition est utilisée sur des structures de représentation polyèdrique. On trouve une utilisation de ce modèle pour le découpage (clipping) de facettes dans [CCV85] et pour les opérations booléennes dans [BN85] [NAB86] [Nav86].

Dans ce modèle, on définit trois nouveaux types de noeuds terminaux:

- type FACE : Un octant de type face est traversé uniquement par une facette du solide.
- type ARETE : Un octant de type arête est traversé par une arête et par les deux facettes qui s'y rattachent.
- type SOMMET : Un octant de type sommet contient un seul sommet et les facettes qui le traversent se rencontrent en ce point.

La frontière d'un objet n'est plus approchée par des octants de taille minimale. L'arrêt de la décomposition s'effectue lorsqu'une configuration *'simple'* a été détectée : octant vide ou plein, ou une seule face, une seule arête ou un seul sommet dans l'octant.
Chaque octant de type FACE, ARETE ou SOMMET contient la représentation par frontière locale du solide. Il sera donc possible de reconstituer l'objet de façon exacte après le traitement que l'on veut effectuer.
Dans cette méthode, les comparaisons se font au niveau de la décomposition, pour la répartition des facettes dans les octants. De plus, dans les octants feuilles de l'arbre, le nombre de facettes est limité :

- 0 pour les octants type NOIR ou BLANC.
- 1 pour les octants type FACE.
- 2 pour les octants type ARETE.
- 3 ou 4 pour les octants type SOMMET (dans [Nav86], l'auteur limite à quatre le nombre de polygones se rejoignant en un sommet).
  Cette restriction est génante lorsqu'on utilise l'approximation polyèdrique de primitives telle que le CONE. Plus grave est la non prise en compte de certains cas particuliers : lorsque des solides se coupent en un sommet, le nombre de polygones se joignant en un sommet ne peut plus être limité a priori.

Le principe du calcul des opérations booléennes entre deux *Octrees exacts* est le suivant :

- parcours synchrone des deux Octrees. Chaque Octree est la subdivision polygonale d'un solide.
- classification entre les nœuds terminaux de chaque Octree.

La complexité globale d'une opération booléenne est proportionnelle à la longueur de la représentation octale de chaque solide. I. Navazo a montré dans [Nav86] que cette longueur dépend linéairement du nombre de sommets et du nombre de polygones présents dans un octant de type SOMMET.

Le principal intérêt des Octrees Polygonaux (ou Exacts) est donc de réaliser les calculs géométriques (coûteux en temps de calcul) à un niveau *local*. Nous allons utiliser la même définition de noeuds terminaux comme critère d'arrêt lors de la subdivision spatiale.

3.2 Les Octrees Booléens

Les Octrees Polygonaux (ou Exacts) ont l'avantage de fournir une méthode de subdivision permettant de délimiter des espaces où le nombre de facettes est limité. La méthode que nous avons utilisée s'inspire de celle_ci. Elle peut être présentée comme la projection d'un arbre CSG dans un Octree exact. C'est-à-dire la projection de l'expression booléenne dans chaque octant. Nous appelerons cette méthode: *Octree Booléen.*

3.2.1 Définitions

Les définitions de différents espaces a pour but de mettre en évidence les volumes dans lesquels les calculs géométriques (classification – sélection) ne seront pas nécessaires.

Espace de définition : l'espace de définition correspond au volume englobant d'un solide (ou d'une primitive). Lors de l'évaluation d'une expression booléenne (représentée par un arbre), toutes les facettes du solide résultant appartiennent à cet espace, il sera donc intéressant de le délimiter afin d'éliminer les calculs hors de cet espace. Si on désire le calculer explicitement, l'espace de définition correspond à l'évaluation de l'arbre CSG sur les volumes englobants des primitives. Il faut cependant apporter une nuance: pour l'opérateur différence, on ne doit pas retirer un volume qui maximise (volume englobant) le volume réel de l'opérande à soustraire, mais un volume qui le minimise (volume englobé).

Espace d'interaction : l'espace d'interaction est défini pour une opération. C'est l'espace commun entre ses opérandes. Quelque soit l'opération, il n'y a pas lieu de poursuivre la subdivision hors de cet espace: on se trouve à l'extérieur de l'espace de définition d'une des opérandes (ou des deux). Le résultat sera alors soit un espace vide, soit la recopie de l'opérande non vide.

Espace de calcul : l'espace de calcul tient compte du niveau global: l'arbre CSG. Pour chaque opération, on le définit comme l'intersection entre l'espace d'interaction de l'opération et l'espace de définition global. C'est dans ce volume que les calculs de classification seront nécessaires. Intuitivement, il signifie que l'interaction de deux solides hors de cet espace n'a pas besoin d'être évaluée.

Le choix réalisé n'est pas le calcul *explicite* de ces espaces à l'aide de structures telles que les volumes englobants. L'utilisation d'un arbre octal permet, de façon adaptative, de se rapprocher de ces volumes. Le niveau de décomposition d'un octant est fonction de la complexité des données y figurant. Par exemple, pour conclure qu'un volume est hors d'un espace d'interaction, il faut trouver l'un des opérandes absent de cet espace. Si on ne peut conclure à un certain niveau de la décomposition octale, on subdivise le volume pour se rapprocher de l'espace réel d'interaction.

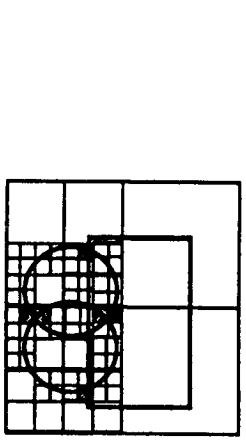


Figure 13 : Exemple 2D de subdivision de l'espace avec un *Octree Exact.*

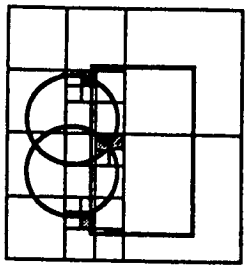


Figure 14 : Exemple 2D de subdivision de l'espace avec un *Octree Booléen.*

3.2.2 La méthode des Octrees Booléens

Le but recherché est double:

1. limiter la décomposition spatiale des primitives aux *espaces d'interaction.*
2. limiter la classification des polygones dans l'*espace de calcul.*

Dans la méthode présentée par I. Navazo (cf [Nav86] [NAB86]), les primitives sont préalablement décomposées en Octrees exacts. Lors de l'évaluation des opérations booléennes, le calcul s'effectue sur deux Octrees et rend comme résultat l'Octree représentant le nouveau solide. L'évaluation de l'ensemble de l'arbre CSG est réalisée de manière ascendante. La Figure 13 présente un exemple de cette méthode.

L'espace de calcul se situe ici au niveau des opérations: on ne tient pas compte du niveau global. Les opérandes seront évaluées systématiquement, même si, pour un espace donné, celles_ci n'interviennent pas. La méthode que nous avons mise en oeuvre peut être représentée à l'aide de l'exemple de la figure 14.

La recherche *implicite* des espaces s'effectue à deux niveaux :

- au niveau d'une opération (espace d'interaction), les octants de type NOIR ou BLANC permettent le remplacement de cette opération par l'une des opérandes (cf tables 16).
- si le résultat de la simplification d'une opération est de type NOIR ou BLANC, la simplification se propage récursivement dans l'arbre. Lorsque dans un octant, l'arbre entier est réduit à une feuille, on se trouve à l'extérieur de l'espace de calcul.

Les structures utilisées sont décrites à l'aide des formules (style BNF) suivantes:

$$
\begin{array}{lll}
\textit{Octree} & ::= & \textit{CSG\_tree} \mid (\textit{Octant}_0, \ldots, \textit{Octant}_7) \\
\textit{Octant}_i & ::= & \textit{Octree} \\
 & & \\
\textit{Arbre\_CSG} & ::= & \textit{Element} \mid (\textit{Operation}, \textit{Fils}_{gauche}, \textit{Fils}_{droit}) \\
\textit{Fils}_x & ::= & \textit{Arbre\_CSG} \\
 & & \\
\textit{Operation} & ::= & \cup, \cap, - \\
 & & \\
\textit{Element} & ::= & \{\textit{Type}, \textit{liste d'Aretes}, \textit{liste de Sommets}, \textit{liste de Polygones}, \\
 & & [\textit{Plan de Construction}]\} \\
\textit{Type} & ::= & \textit{NOIR} \mid \textit{BLANC} \mid \textit{GRIS} \mid \textit{FACE} \mid \textit{ARETE} \mid \textit{SOMMET}
\end{array}
$$

Un Octree Booléen est décomposé en octants de façon classique. Chaque octant contient la projection de l'arbre CSG dans le volume concerné. La projection de l'arbre CSG dans un octant est l'arbre auquel on enlève les primitives dont les faces n'interviennent pas dans l'octant. Les simplifications sont alors fonction de l'opération (cf tables 16). Chaque arbre est composé des primitives dont la frontière traverse l'octant. Pour chaque primitive, il contient la description des données (arêtes, sommets, polygones) intervenant dans l'octant. Lorsqu'un arbre est réduit à un élément, cela signifie que pour l'octant correspondant on a la connaissance des données constituant le solide à calculer: l'évaluation est terminée dans cet espace. Dans le cas contraire, la subdivision se poursuit et les calculs seront effectués, lorsque des configurations terminales pour les éléments auront été atteintes (élément de type FACE, ARETE ou SOMMET). Les éléments de type NOIR ou BLANC, contribuent à la simplification de l'arbre. Suivant l'opération (binaire), si un des opérandes est de type NOIR ou BLANC, le sous_arbre correspondant est remplacé soit par l'autre opérande (ou son complémentaire), soit par un élément de type NOIR ou BLANC (cf tables 16).

3.2.3 Mise en oeuvre des Octrees Booléens

La subdivision de l'espace et le calcul des opérations (classification – sélection) sont entrelacées dans la mise en oeuvre des Octrees Booléens.

- La subdivision intervient pour réduire l'espace de calcul et fait appel à l'évaluation des opérations booléennes lorsqu'une configuration *simple* est détectée.
- Le calcul des opérations booléennes est effectué dans un espace (Octant) dans lequel les deux opérandes ont une configuration reconnue: type FACE, ARETE ou SOMMET.
- Après les calculs géométriques (classification – sélection), si le résultat est de type GRIS, l'octant est subdivisé.
- Le traitement *subdivision–calcul* se termine lorsqu'il n'y a plus d'arbre dans l'Octree Booléen (en d'autres termes, lorsque toutes les projections de l'arbre CSG dans les

octants sont réduites à un Elément: description des arêtes, sommets et polygones du solide calculé).

La procédure principale réalise le traitement à partir d'une expression booléenne (représentée par un arbre binaire) sur des polyèdres de la façon suivante :

- appel à la procédure *Subdivise* pour construire l'*Octree exact* correspondant à l'expression booléenne.
- parcours de cet Octree afin de trouver l'ensemble des facettes constituant le solide évalué.

L'espace de subdivision (espace englobant de l'arbre CSG) est 'normalisé' : les coordonnées des sommets sont placées dans l'intervalle $0 \leftrightarrow 1024$. Ceci est réalisé par une *affinité* [1] et une *translation*. Ce calcul est effectué sur les matrices de positionnement (de chaque primitive) dans la scène . Au premier appel de la procédure, les éléments de l'arbre sont de type GRIS.

La procédure *Subdivise* : la procédure *Subdivise* est récursive: à un niveau de récursivité correspond un niveau de subdivision spatiale.

Lors de l'appel à *Subdivise* les paramètres transmis sont :

- une expression booléenne (Arbre) d'éléments représentant un solide ou une partie de solide (projection du solide dans l'octant).
- la description de l'espace de subdivision (octant). Un octant est défini à l'aide des coordonnées du point d'origine (x,y,z) et de sa taille (scale).

La procédure *Subdivise* va découper l'octant transmis lors de l'appel en huit sous_cubes. Le calcul des coordonnées des origines des sous_octants est effectué à l'aide des vecteurs VX, VY et VZ. Pour chaque sous_octant, la procédure construit la projection de l'arbre dans ce sous_espace (réalisé par la procédure *Cons_arbre*). Ce nouvel arbre (arbre1) est alors analysé et réduit si cela est possible (réalisé par la procédure *Réduire_arbre*). Enfin, si cet arbre est réduit à un élément, il n'y a pas lieu de continuer la subdivision: c'est la condition d'arrêt de la récursivité. Sinon, le sous_espace est subdivisé à son tour.

Le résultat de la procédure *Suddivise* est un *Octree Booléen* dans lequel chaque feuille est réduite à la description des arêtes, sommets et polygones (structure *Elément*) du solide évalué intervenant dans cet espace (octant).

L'énoncé de l'algorithme est le suivant :

[1] Changement d'échelle différent suivant les trois dimensions.

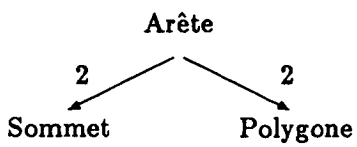
appel: Subdivise (0, 0, 0, 1024, arbre)

Vecteurs servant au calcul de l'origine des sous_octants.

VX = { 0, 1, 0, 1, 0, 1, 0, 1 }
VY = { 0, 0, 0, 0, 1, 1, 1, 1 }
VZ = { 1, 1, 0, 0, 1, 1, 0, 0 }

```
procedure Subdivise ( x, y, z, scale, arbre ) resultat Octree
| sq := scale / 2 ;
|
| pour i=0 jqa 7 faire
|
|   | x1 := x + VX[i] * sq ;
|   | y1 := y + VY[i] * sq ;
|   | z1 := z + VZ[i] * sq ;
|   |
|   | arbre1 := cons_arbre(x1,y1,z1,sq,arbre) ;
|   |
|   | réduire_arbre (arbre1) ;
|   |
|   | si arbre1 est une FEUILLE alors
|   |   | Octree.fils_i := arbre1.élément ;
|   |   sinon
|   |   | Octree.fils_i := Subdivise(x1,y1,z1,sq,arbre1) ;
|   |   finsi
|   fait
fin
```

La procédure *Cons_arbre* : Pour chaque élément de l'arbre transmis à l'appel, la procédure *Cons_arbre* recopie les sommets, arêtes et polygones intervenant dans le nouveau sous_espace (octant). L'organisation choisie pour les données (sommets, arêtes,polygones) est la structure symétrique :



Une étude comparative des différentes organisations a été réalisée par T. C. Woo ([Woo85]): la structure symétrique y apparait comme le meilleur compromis entre les critères :

- d'accès aux informations;
- d'occupation mémoire.

Par rapport aux autres structures classiques:

- meilleur accès aux informations que la structure linéaire suivante :
  - objet = ensemble de faces.
  - face = ensemble de polygones.
  - polygone = ensemble d'arêtes.
  - arête = couple de sommets.
- plus faible occupation mémoire que la structure de données de Baumgart (*Winged_edge*): dans la structure de Baumgart, les arêtes pointent sur les sommets début et fin, sur les deux faces adjacentes (gauche ou droite) et sur les quatre arêtes adjacentes de ces deux faces.

Du fait de la structure choisie, les tests de la procédure *Cons_arbre* sont effectués en premier lieu sur la liste des arêtes :

1. pour chaque arête, déterminer si l'une des extrémités appartient à l'octant.
2. si les deux sommets d'une arête sont à l'extérieur d'un octant, on regarde si cette arête coupe l'octant.

   Pour ces deux premiers tests, on utilise un codage des sommets du type SUTHERLAND et SPROULL (cf [SS68]):
   Un code de six bits indique pour chaque sommet sa situation par rapport aux six demi-espaces définis par chaque plan de l'octant.
   Pour un octant *fils* (niveau de décomposition i), les codes peuvent être en partie hérités, puisque chaque octant possède trois plans communs avec l'octant *père* (niveau de décomposition $i-1$). Ceci permet de minimiser les opérations de comparaison dans le codage des sommets.

   - Le test (1) consiste à regarder si le code d'un des sommets de l'arête est égal à *000000*.
   - Test (2) : si l'arête n'appartient pas à un demi_espace externe (ET logique sur les les codes des deux sommets égal à *000000*), alors on calcule le point d'intersection entre l'arête et les plans de l'octant séparant les deux sommets. Si un point d'intersection appartient à la frontière de l'octant, l'arête coupe l'octant et doit être recopiée.

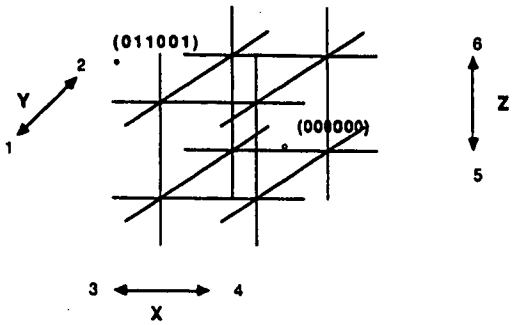


Figure 15 : Codage des sommets dans l'espace: codage du type SUTHERLAND & SPROULL.

Lorsque l'on recopie une arête, on recopie également les sommets et les polygones associés.

Les tests suivant consistent à examiner les polygones restants. N'ayant pas d'arête dans l'octant, deux cas de figure peuvent se présenter pour chaque polygone:

- soit il est externe à l'octant;
- soit il *traverse* tout l'octant.

Les tests (3) et (4) sont effectués sur les polygones:

3. le plan support du polygone coupe_t'il l'octant ?
   Ceci est réalisé en étudiant la situation des extrémités des diagonales de l'octant vis-à-vis du plan.

4. Si oui, on regarde si le point d'intersection *plan–diagonale* est interne au polygone. Les polygones étant convexes, l'algorithme choisi pour la classification *point–facette* est celui de DAHAN et LE TUAN (cf [Heg85]).

Ce dernier test est le plus coûteux en temps de calcul. L'algorithme de DAHAN et LE TUAN consiste à calculer la distance signée du point à chaque arête. Dès le premier changement de signe entre deux distances, on peut conclure que le point est externe.

A la fin de la recopie des différentes données, il reste à déterminer le type de l'octant :

- aucun polygone dans le nouvel octant: le type est NOIR ou BLANC. La classification d'un point de l'octant *fils* (le point origine par exemple) par rapport aux plans présents dans l'octant *père* permet de différencier les deux possibilités (pour ce faire, on utilise l'information *plan de construction* défini dans 4.1).
- un seul polygone et aucune arête : type FACE.

- une seule arête et les deux polygones associés : type ARETE.
  Après l'évaluation d'une opération booléenne, plusieurs arêtes peuvent avoir les mêmes extrémités (elles appartiennent à un même chaînage : cf 4.3). Elles sont considérées alors comme une seule arête.
- toutes les arêtes présentes ont une extrémité commune et il n'y a pas de polygone englobant l'octant : type SOMMET.
- autrement : type GRIS.

La procédure *Réduire_arbre* : la procédure *Réduire_arbre* réalise une simplification récursive de l'arbre. Lorsqu'un élément de type NOIR ou BLANC est présent, celui_ci participe à cette simplification (Fig. 16). Au niveau d'une opération, si les deux éléments sont de type FACE, ARETE ou SOMMET, la procédure *Réduire_arbre* fait appel à l'évaluation de l'opération booléenne (cf section 4).

Union A ∪ B

A	A ∪ B	B	A ∪ B
NOIR	NOIR	NOIR	NOIR
BLANC	B	BLANC	A

Intersection A ∩ B

A	A ∩ B	B	A ∩ B
NOIR	B	NOIR	A
BLANC	BLANC	BLANC	BLANC

Difference A - B

A	A − B	B	A − B
NOIR	¬ B	NOIR	BLANC
BLANC	BLANC	BLANC	A

Figure 16 : Simplification des arbres à l'aide des éléments de type NOIR ou BLANC

Pour une question de lisibilité les procédures *Cons_arbre* et *Réduire_arbre* ont été séparées; mais elles peuvent être mixées afin de ne pas effectuer les tests de recopie pour

un élément qui sera éliminé lors de la simplification. Par exemple, si l'opérande gauche d'une *intersection* est de type BLANC, il est inutile de construire l'opérande droit.

Nous avons vu comment les simplifications interviennent pour approcher l'espace de calcul. Dans la section suivante, nous allons décrire la mise en oeuvre du calcul effectif des opérations booléennes. En particulier, nous allons étudier les problèmes de cohérence des données entre les différents octants.

4 Calcul des opérations booléennes

Dans la section 2 , nous avons présenté le principe de l'évaluation des opérations booléennes sur les polyèdres :

1. éclatement des polygones des deux solides opérandes ;
2. classification de ces polygones vis-à-vis de l'autre solide ;
3. sélection des polygones suivant l'opération (cf tables Fig. 16).

Pour réaliser (1), on introduit une notion d'*historique* des polygones éclatés. Pour (2), on définit un *plan de construction* rattaché aux octants. En premier lieu, nous allons présenter la méthode de classification mise en oeuvre, puis le choix effectué pour le découpage des polygones.

4.1 Classification des polygones

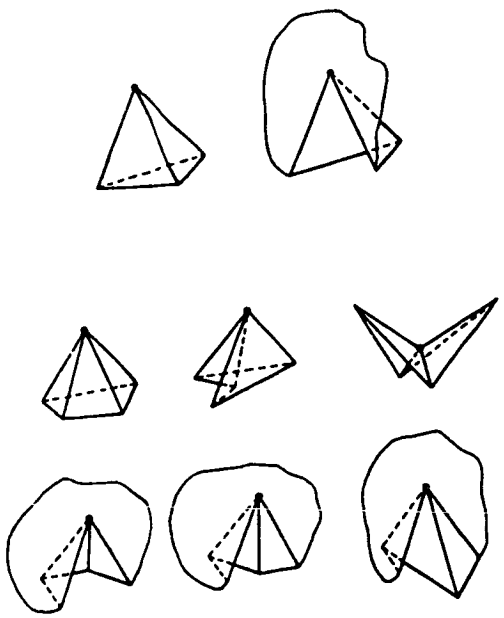


Figure 17 : Configurations possibles des sommets avec trois ou quatre faces. Pour chacune d'elles le solide peut appartenir à l'un des demi_espaces définis par les faces.

D.H. Laidlaw utilise une technique style 'lancer_de_rayon' pour réaliser la classification *Polygone_A*↔*Solide_B*. Le rayon créé a pour origine le barycentre du polygone et comme direction la normale au plan support. Pour évaluer la position du polygone, on calcule le point d'intersection le plus proche entre le rayon créé et l'ensemble des facettes du *Solide_B*.

- Si aucun *Polygone_B* n'est recontré, le *Polygone_A* est EXTERNE au *Solide_B*.

- Sinon, le *Polygone_B* le plus proche est retenu, et on calcule un point de rencontre entre les droites normales des deux polygones. Si celui_ci est interne au *Polygone_B*, le *Polygone_A* est sur la frontière du *Solide_B*. Le type sera IDEM ou OPPOSÉ suivant l'orientation du *Polygone_B*. Sinon, il sera INTERNE ou EXTERNE suivant l'orientation du *Polygone_B*.

Cette technique de classification est très coûteuse en temps de calcul, surtout lorsqu'on n'utilise pas de subdivision de l'espace.

I. Navazo ([Nav86]) réalise la classification (dans un Octree exact) à l'aide d'un codage des différentes configurations possibles des plans. Une configuration correspond à une expression booléenne sur les demi_espaces délimités par les plans. Le codage est réalisé en étudiant la convexité du solide au voisinage des arêtes. Afin de réaliser un codage exhaustif des différents cas de figure, le nombre de polygones se joignant en un sommet est limité à quatre. Pour les octants de type SOMMET, et avec cette limite, il y a douze configurations possibles (cf Fig.17). Nous avons vu dans 3.1.3 que cette limite ne permet pas la prise en compte de certaines configurations particulières (solides se coupant en un sommet).

Nous avons opté pour une méthode plus générale permettant de résoudre les cas particuliers. Dans cette méthode, la classification est réalisée à l'aide d'un ***plan de construction*** du solide au niveau de chaque octant. Un ***plan de constuction*** est une expression booléenne sur les demi_espaces concernés. Cette expression se présente sous la forme d'un arbre dans lequel les feuilles sont des demis_espaces et les nœuds des opérateurs booléens ($\cup, \cap, -$ et la négation $\neg$). Le but est de décrire, dans chaque octant, la configuration d'un solide sous forme d'une réunion de parties convexes. Une partie convexe est définie comme l'intersection de demi_espaces (un demi_espace est représenté à l'aide d'un plan orienté).

La classification locale d'un *Polygone*A vis-à-vis d'un *Solide*B est réalisé à l'aide d'un ***plan de construction***. La procédure de découpage des polygones, dépendant de ce schéma de classification, sera décrit par la suite (cf 4.2). Après l'appel de la procédure de découpage, les *Polygones*A et *Polygones*B (polygones des deux opérandes A et B) ne se coupent plus, dès lors la classification *Polygone*A/*Solide*B est évaluée à l'aide de classifications de type *Point*A/*ConvexB*$_i$ où le *Point*A est un point local interne au *Polygon*A et *ConvexB*$_i$ une partie convexe du *Plan de Construction* du *Solide*B. La classification *Point*A/*Convex B*$_i$ utilise des classifications du type *Point*A/*Demi_espaceB*$_{i_j}$, où *Demi_espaceB*$_{i_j}$ est un demi_espace de la partie convexe *ConvexB*$_i$. Le parcours d'un *Plan de Construction* est effectué comme suit :

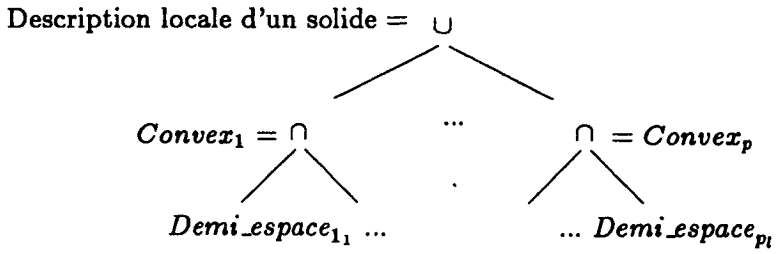


Figure 18 : Un *plan de construction* est une réunion de parties convexes.

Etant donné un $PointA \in PolygoneA$

$$(\exists\, j,\ PointA\ EXTERNE\ Demi\_espaceB_{i_j}) \implies (PointA\ EXTERNE\ ConvexB_i)$$

$$(\forall\, j,\ PointA\ INTERNE\ Demi\_espaceB_{i_j}) \implies (PointA\ INTERNE\ ConvexB_i)$$

$$(\exists\, j,\ PointA\ FRONTIERE\ Demi\_espaceB_{i_j}) \quad \text{et}$$
$$(\forall\, k \neq j,\ PointA\ INTERNE\ Demi\_espaceB_{i_k}) \implies (PointA\ FRONTIERE\ ConvexB_i)$$

$$(\exists\, i,\ PointA\ INTERNE\ ConvexB_i) \implies (PointA\ INTERNE\ SolidB)$$
$$\implies (PolygonA\ INTERNE\ SolidB)$$

$$(\forall\, i,\ PointA\ EXTERNE\ ConvexB_i) \implies (PointA\ EXTERNE\ SolidB)$$
$$\implies (PolygonA\ EXTERNE\ SolidB)$$

$$(\exists\, i,\ PointA\ FRONTIERE\ ConvexB_i) \quad \text{et}$$
$$(\forall\, k \neq i,\ PointA\ EXTERNE\ ConvexB_k) \implies (PointA\ FRONTIERE\ SolidB)$$
$$\implies (PolygonA\ FRONTIERE\ SolidB)$$

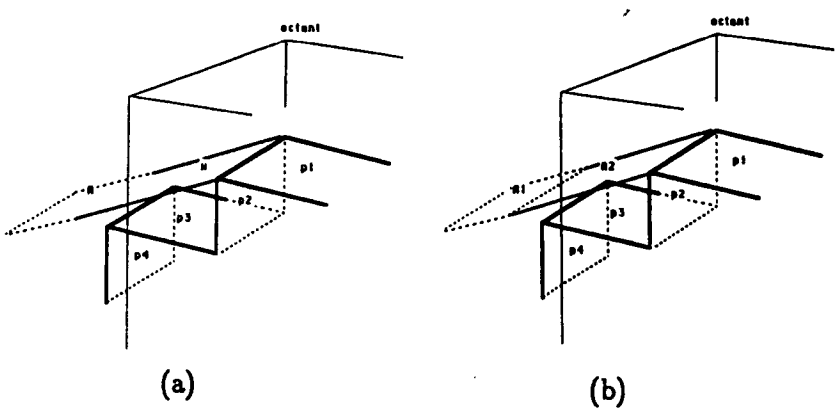
Pour un *Polygone*A de type FRONTIÈRE, on différencie (type IDEM ou OPPOSÉ) en comparant l'orientation du plan support du polygone et celle du plan coplanaire dans le *Plan de Construction.*

La structure *plan de construction* permet de décrire toutes les configurations possibles d'un solide au niveau d'un octant. L'élaboration des *plans de construction* n'est faite qu'au niveau du calcul des opérations sur les noeuds terminaux. La contrainte résultante est que les primitives solides doivent être convexes . Dans un octant, si le *plan de constuction* n'est pas présent, cela signifie que les plans forment une partie convexe. De cette manière, le nombre d'éléments dans un plan de construction est limité au nombre maximum de plans présents dans deux noeuds terminaux. Après chaque évaluation locale d'une opération, le

$$
\begin{array}{ll}
\cap(\cup(x,y),\, z) & \longrightarrow \cup(\cap(x,z),\, \cap(y,z)) \\
\cap(x,\, \cup(y,z)) & \longrightarrow \cup(\cap(x,y),\, \cap(x,z)) \\
\cup(\cup(x,y),\, z) & \longrightarrow \cup(x,\, \cup(y,z)) \\
\cap(\cap(x,y),\, z) & \longrightarrow \cap(x,\, \cap(y,z)) \\
\neg(\cup(x,y)) & \longrightarrow \cap(\neg(x),\, \neg(y)) \\
\neg(\cap(x,y)) & \longrightarrow \cup(\neg(x),\, \neg(y)) \\
-(x,y) & \longrightarrow \cap(x,\, \neg(y)) \\
\neg(\neg(x)) & \longrightarrow x
\end{array}
$$

Figure 19 : Système de réécriture pour les *plans de construction*. Les opérateurs $\cap$, $\cup$ et $-$ sont des opérateurs binaires, l'opérateur $\neg$ (négation) est unaire. x, y et z sont des expressions booléennes de demi_espaces.

Plan de Construction est mis sous forme canonique à l'aide de règles de réécriture héritées de l'algèbre booléenne (cf Fig. 19).



(a) (b)

Figure 20 : Problèmes de classification à un niveau local. P1, P2 et P3 sont des polygones du *solideB* interne à l'octant. P4 est externe à l'octant. Deux méthodes de classification du *polygoneA* vis-à-vis de la description locale du *solideB*.

Remarques concernant l'aspect local de la classification Lorsque l'on travaille au niveau d'un octant, on dispose d'une description locale des solides (à l'aide des *plans de construction*). Pour que la classification *polygoneA*$\leftrightarrow$*solideB* soit valide il faut respecter l'une des conditions suivantes :

- prendre un point du *polygoneA* situé dans l'octant. C'est-à-dire trouver un point commun entre le polygone et l'octant (exemple, le point x de la figure 20 a). Pour ce

faire on doit comparer le polygone aux plans frontière de l'octant. C'est une méthode trop coûteuse pour être satisfaisante. De plus, les cas particuliers des polygones n'intersectant que les plans frontière de l'octant nécessitent un traitement particulier.

ou

- réaliser la classification sur des *sous_polygonesA* dont tous les points internes ont la même situation vis-à-vis du *solideB*. Pour ce faire, on découpe tous les *polygoneA* par chaque plan support des polygones du *solideB* (dans la figure 20 b le *polygoneA* est découpé par le plan support du *polygone*P3). Avec ce principe, tous les points internes d' un *sous_polygoneA* ont la même situation vis-à-vis de la description locale du *solideB*. Le résultat de la classification locale d'un sous_polygone peut apparaître en contradiction avec la réalité globale; cela ne peut arriver qu'avec des sous_polygones externes à l'octant qui seront éliminés de cet octant quelque soit leur type de classification (exemple *sous_polygone*A1 de la figure 20 b).

Après la classification, les polygones sont sélectionnés en fonction du type de l'opérateur booléen (cf tables Fig. 16).

4.2 Découpage en facettes convexes

Le fait de fournir comme résultat un polyèdre formé de facettes convexes est un atout supplémentaire pour la performance des traitements ultérieurs : les algorithmes (d'affichage, d'élimination des parties cachées ...) sont plus rapides avec des polygones convexes. Le principe de l'algorithme, inspiré de la méthode de D.H. Laidlaw ([LTH86]), est de calculer, pour tout couple de polygones se coupant, le segment intersection (qui peut être réduit à un sommet); et en fonction de ce segment, éclater les deux polygones en sous_polygones ne se coupant plus. Le critère d'intersection entre deux polygones peut_être précisé comme suit : deux polygones non_coplanaires se coupent si et seulement si un segment de l'un (interne ou frontière) se trouve à l'intérieur de l'autre. Les polygones coplanaires et les polygones partageant une arête ou un sommet ne se coupent pas suivant ce critère. Chaque polygone intersecté sera éclaté par le polygone intersectant.

4.2.1 Présentation des calculs

Calcul d'un segment intersection entre un polygone et un plan : on calcule les distances signées de chaque sommet au plan,

- Si la distance est nulle, le sommet est dans le plan;
- Si une arête coupe le plan (les extrémités ont des distances de signes opposés), le calcul des coordonnées du point d'intersection est effectué en évaluant le rapport des deux distances (cf Fig. 21).

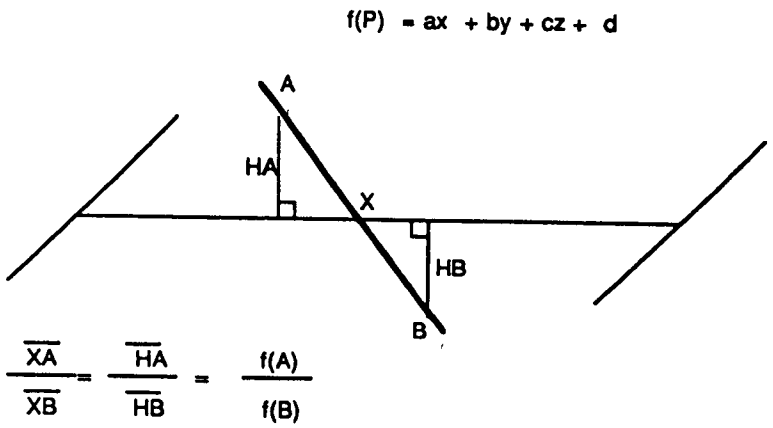
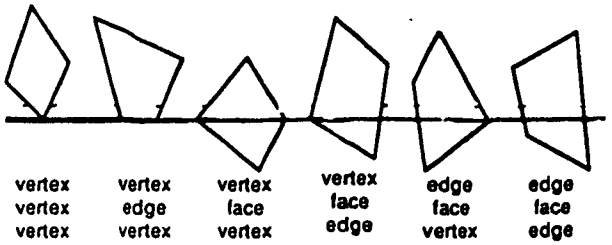


Figure 21 : Calcul du point d'intersection entre un segment et un plan.

Il existe six types d'intersections possibles :



Parmi ces six types d'intersections, seuls les quatre derniers entrainent un découpage du polygone.

Calcul de l'intersection entre un *polygone A* et un *polygone B* : en premier lieu, on calcule le segment d'intersection de chaque polygone (A et B) avec le plan support de l'autre polygone : on obtient *segment A* et *segment B*.

Le type de l'intersection est conservé dans un descripteur à trois champs:

- type du point origine du segment ;
- type des points internes au segment ;
- type du point d'arrivée du segment ;

Le type d'un point peut être, suivant que ce point rencontre un sommet, une arête ou l'intérieur du polygone :

- Sommet (Vertex) ;
- Arête (Edge) ;
- Face (Face) ;

Si un *segmentA* (resp. *segmentB*) n'est pas réduit à un seul sommet (type des points internes égal à Sommet) ou n'est pas vide, le *polygoneA* est découpé à l'aide de ce segment qui devient une arête des *sous_polygones* A1 et A2 ainsi créés. Ensuite, on évalue l'intersection entre le *segment A* et le *segment B* pour obtenir *segmentA_in_segmentB* et *segmentB_in_segmentA* qui ne diffèrent que par leur descripteur de type (cf Fig.22).

Si l'une des extrémités du *segmentA_in_segmentB* (resp. *segmentB_in_segmentB*) est modifiée par rapport au *segment A* (resp. *segment B*), alors le type du nouveau point est la recopie du type des points internes au *segment A* (resp. *segment B*)(cf Fig.22). L'arête concernée dans les *sous_polygones* A1 et A2 est éclatée a l'aide du *segmentA_in_segmentB*.

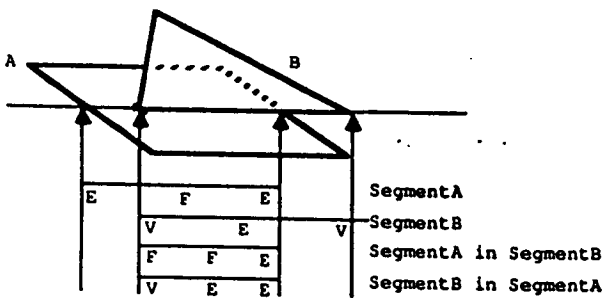


Figure 22 : Intersection de deux segments (tiré de [LTH86]).

4.3 Cohérence entre données globales et découpage local

Au cours du traitement, un polygone peut être l'objet de plusieurs éclatements. Pour que le découpage en polygones convexes soit valide, il faut qu'un polygone ne soit effectivement éclaté qu'une seule fois; et par la suite, ce sont ses sous_polygones qui seront éclatés. Pour ce faire, on chaîne à chaque polygone éclaté les deux sous_polygones résultants. Cette structure permet :

- d'assurer la cohérence des polygones entre deux octants différents.
- de faciliter le traitement d'éclatement d'un polygone : à l'aide du chaînage, on accède facilement aux polygones non encore découpés.
- de conserver la liste des polygones déjà comparés, afin de ne pas réaliser plusieurs fois les mêmes calculs dans des octants différents.
- d'invalider un découpage inutile, lorsque après classification, deux *sous_polygones* A1 et A2 appartiennent au même ensemble de classification (*AinB*, *AoutB*, *AonB IDEM* ou *AonB OPPOSÉ*).

Lors du découpage des polygones, on doit également maintenir la cohérence des informations concernant les arêtes. Dans la structure de données choisie (cf 3.2.3), les arêtes relient les polygones et les sommets. Pour une nouvelle arête créée, il faut trouver les deux polygones à relier. Parmi ces nouvelles arêtes, on doit distinguer deux catégories :

- les arêtes frontière : ce sont les segments d'intersection de deux polygones.
- les arêtes internes : elles sont propres à un polygone et sont créées pour assurer la convexité des sous_polygones résultant de l'éclatement du polygone original.

Déterminer, pour une arête interne, quels sont les polygones concernés, ne pose pas de problème particulier. Par contre, pour une arête frontière, qui peut relier deux polygones

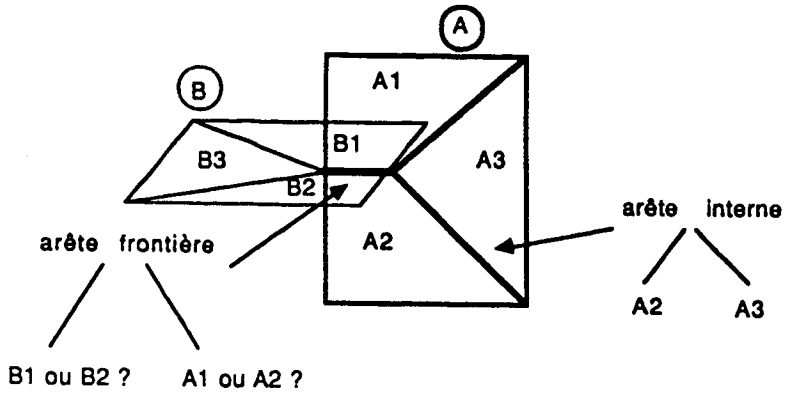


Figure 23 : Problème des arêtes frontière.

d'objets différents, l'étude du voisinage de l'arête s'impose (cf figure 23).

La solution retenue pour résoudre ce problème de cohérence est la suivante :

1. Lors de l'éclatement d'une arête (à l'aide de *segmentAinB* et de *segmentBinA*), les arêtes frontière (identiques) de chaque polygone sont chaînées. On crée une arête en tête de ce chaînage pour recevoir les polygones retenus lors de leur classification.
2. Un polygone retenu après classification se marque dans la structure de données de l'arête en tête du chaînage et retient celle_ci comme l'une de ses arêtes.

Deux cas de figures peuvent se produire :

1. soit deux polygones sont retenus après la classification : dans ce cas l'arête de tête de chaînage devient l'arête valide pour ces deux polygones.
2. soit les quatre polygones sont retenus : dans ce cas la structure de données des arêtes frontière est la même qu'avant le traitement.

Les cas particuliers plus complexes où plusieurs solides se coupent sur une arête sont traités de la même manière à l'aide de ce chaînage sur les arêtes frontière. Le chaînage sur les arêtes identiques est utilisé dans la procédure *cons_arbre* (cf 3.2.3) et permet de reconnaître des cas particuliers d'octants de type ARETE. Les arêtes du chaînage qui ne sont plus valides dans un octant donné sont conservées. Leur contenu peut être nécessaire dans un autre octant.

5 Quelques résultats

Cette méthode a été implémentée à l'IRISA en langage C sur un SUN3/160 utilisant le système Unix. La description CSG est effectuée à l'aide de LGRC [BAP86], qui est un langage de modélisation géométrique développé à l'IRISA. Quelques résultats sont présentés

à l'aide des figures 24 25 26. Ces exemples ont pour but d'illustrer les motivations de cette étude. Les temps CPU pourrait être améliorés, nous ne prétendons pas avoir une implémentation optimale de notre algorithme. La figure 25 illustre une situation particulière où deux primitives se coupent sur une arête commune. Un autre exemple concerne une étude comparative du temps pris par la subdivision spatiale par rapport au temps pris par l'évaluation. La figure 26 donne les temps CPU d'une éxécution selon le niveau maximum de subdivision utilisé. Une subdivision trop importante de l'espace ne donne pas un résultat optimal, la redondance des tests (regarder pour deux polygones s'ils ont déjà été comparés dans les octants voisins) augmente le temps d'évaluation. Dans cet exemple, le temps imparti à la subdivision augmente de façon logarithmique, par contre le temps imparti à l'évaluation décroit de façon exponentielle. L'exemple 24 illustre le comportement linéaire de notre algorithme. Dans la même situation (la réunion de deux sphères), nous avons fait varier le nombre de polygones en paramètrant l'approximation polyèdrique de la sphère.

6 Conclusion

Nous avons présenté une nouvelle méthode d'évaluation d'arbres CSG composés de primitives polyédriques. Notre implémentation actuelle utilise des polyèdres convexes, cependant l'utilisation de polyèdres concave peut s'envisager avec l'ajout d'une information de convexité au niveau des arêtes. Nous avons choisi une subdivision octale plutôt qu'une hiérarchie de boîtes englobantes afin d'obtenir une localisation dynamique plus pertinante des *volumes minimaux de calcul.*

Bibliographie

[ABJN85] D. Ayala, P. Brunet, R. Juan, and I. Navazo. Object representation by means of non minimal division quadtrees and octrees. *ACM transactions on graphics*, 4(1), 1985.

[BAP86] K. Bouatouch, B. Arnaldi, and Thierry Priol. LGRC: un langage pour la synthèse d'image par lancer de rayon. *T.S.I.*, 5(6), 1986.

[BN85] P. Brunet and I. Navazo. *Geometric Modelling using Exact Octree Representation of Polyhedral Objects*, page 159. EUROGRAPHICS'85, Nice France, 1985.

[CCV85] I. Carlbom, I. Chakravarty, and D. Vanderschel. A hierarchical data structure for representing the spatial decomposition of 3D objects. *IEEE Computer graphics and applications*, april 1985.

[Heg85] G. Hégron. *Synthèse d'image: algorithmes élémentaires.* Dunod informatique, août 1985.

[KSY85] T.L. Kunii, T. Satoh, and K. Yamaguchi. Generation of topological boundary representations from octree encoding. *IEEE Computer graphics and applications*, 5(3), march 1985.

[LTH86] D.H. Laidlaw, W.B. Trumbore, and J.F. Hughes. Constructive solid geometry for polyhedral objects. *Communication of the ACM*, 20(4), august 1986.

[MS82] M. Mäntylä and R. Sulonen. GWB: a solid modeler with euler operators. *IEEE Computer graphics and applications*, september 1982.

[NAB86] I. Navazo, D. Ayala, and P. Brunet. *A Geometric Modeller Based on the Exact Octtree Representation of Polyhedra.* Computer Graphics Forum5, 1986.

[Nav86] I. Navazo. *Contribució a les tècniques de modelat geomètric d'objectes polièdrics usant la codificació amb arbres octals.* PhD thesis, Universitat politècnica de Catalunya, Barcelone, 1986.

[Req80] A.A.G. Requicha. Representations for rigid solids: theory, methods, and systems. *Computing Surveys*, 12(4), december 1980.

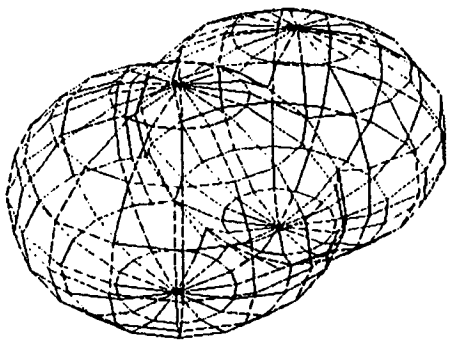
[RT78] A.A.G. Requicha and R.B. Tilove. *Mathematical Foundations of Constructive Solid Geometry: General Topology of Regular Closed Sets.* Production automation project 27, University of Rochester, New_York, march 1978.

[RV85] A.A.G. Requicha and H.B. Voelcker. Boolean operations in solid modeling: boundary evaluation and merging algorithms. *Proceedings of the IEEE*, 73(1), january 1985.

[SS68] I.E. Sutherland and R.F. Sproull. *A clipping divider*, page 765. FICC Thompson books, Washington, D.C., 1968.

[Til80] R. B. Tilove. Set membership classification: a unified approach to geometric intersection problems. *IEEE Transactions on computers*, 29(10), october 1980.

[Woo85] T. C. Woo. A combinational analysis of boundary data structure shemata. *IEEE Computer graphics and applications*, 5(3), March 1985.



Niveau de représentation de la sphère	0	1	2	3	4
Nombre de polygones des primitives	16	64	256	1024	4096
Nombre de polygones résultant	19	62	238	922	3544
Temps CPU pour la subdivision	1.0	3.3	8.5	33.6	124.4
Temps CPU pour l'évaluation	0.6	1.7	4.9	10.4	21.3
Temps CPU total	1.6	5.0	13.4	44.0	145.7

Figure 24 : Un exemple de temps d'éxécution (en sec.) dépendant du nombre de polygones en présence.

Temps CPU pour la subdivision	0.2 sec.
Temps CPU pour l'évaluation	0.6 sec.

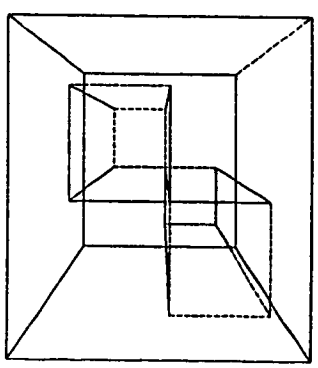
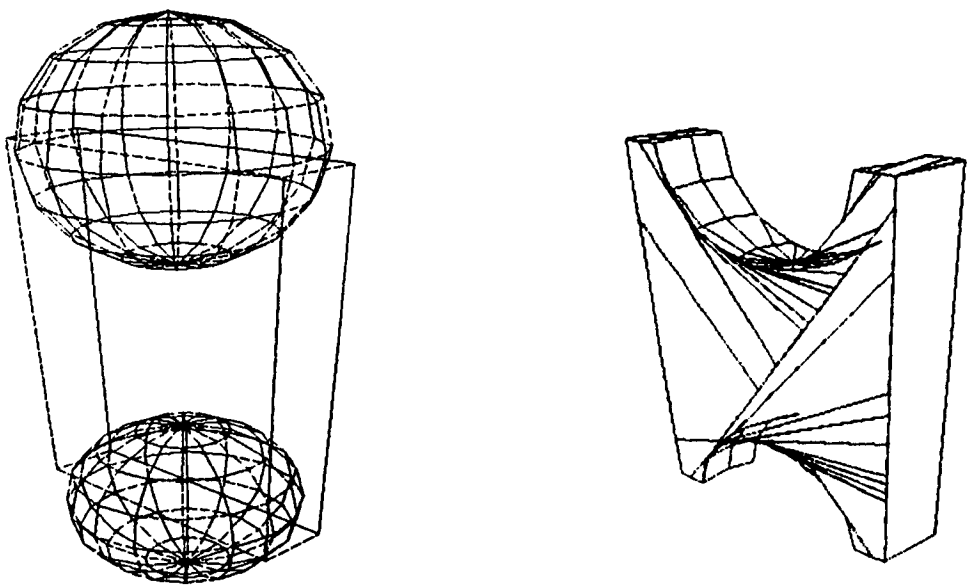


Figure 25 : Une situation particulière.



Niveau de subdivision	0	1	2	3	4	5	6	7	8	9	10
Temps CPU pour la subdivision	0	2.2	5.5	11.5	15.0	21.2	23.9	24.1	25.0	25.5	25.6
Temps CPU pour lévaluation	1535	362.3	92.9	24.7	32.8	37.2	38.5	39.0	38.7	34.1	35.1
Temps CPU total	1535	364.5	98.4	36.2	47.8	58.4	62.4	64.6	63.7	59.6	60.7

Figure 26 : Un exemple d' éxécution dépendant du niveau maximal de subdivision de l'espace.

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique