



**HAL**  
open science

# Segmentation dynamique d'images tridimensionnelles

François Leitner

► **To cite this version:**

François Leitner. Segmentation dynamique d'images tridimensionnelles. Modélisation et simulation. Institut National Polytechnique de Grenoble - INPG, 1993. Français. NNT: . tel-00344080

**HAL Id: tel-00344080**

**<https://theses.hal.science/tel-00344080>**

Submitted on 3 Dec 2008

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# THESE

présentée par  
**François LEITNER**

pour obtenir le titre de

**Docteur de l'Institut National Polytechnique de Grenoble**  
**SPECIALITE INFORMATIQUE**

## **Segmentation dynamique d'images tridimensionnelles**

Soutenue le 27 septembre 1993 devant la commission d'examen

Jacques DEMONGEOT :	<i>Président</i>
Henri MAITRE,	
Michel PIERRE :	<i>Rapporteurs</i>
Philippe CINQUIN,	
Roger MOHR,	
Claude PUECH :	<i>Examineurs</i>



## RESUME

Ce travail traite de la segmentation d'images médicales : ce type de problème est difficile à résoudre à cause de la piètre qualité du signal, et de la fréquente nécessité d'avoir une connaissance spécifique. Aussi se restreint-on à des objets de type volumique courants, tels que les os ou des organes massifs. On modélise une surface fermée à l'aide de produits tensoriels de B-splines (utilisés en CAO), et on propose deux formes de représentation : soit un cylindre à pôles, défini par une seule grille de contrôle, soit un graphe de contrôle, reliant continûment plusieurs grilles. Ces représentations sont adaptées aux déformations globales ou locales, et aux changements de topologie. La déformation d'une telle surface est modélisée par le dégonflage d'un ballon : partant d'une forme englobant la scène, on laisse se dégonfler la surface pour atteindre successivement l'enveloppe convexe, puis l'enveloppe localement convexe, et enfin la surface externe de la scène. L'apport principal de cette méthode consiste en la possibilité d'appréhender un objet complexe, par sa topologie, ou par le nombre de ses composantes connexes. La méthode de segmentation utilise alors cette technique sur l'ensemble des indices de contours détectés dans les images. On peut voir cette méthode comme un outil modélisant l'interpolation naturelle de l'oeil. Ce travail se place ainsi parmi les opérateurs de niveau intermédiaire, en apportant la mise en cohérence de certaines propriétés géométriques de la fonction densité associée aux images. Ceci ne résoud cependant pas le problème de connaissance *a priori*, une interface utilisateur pouvant y remédier dans les cas critiques.

## ABSTRACT

This work is about segmentation of medical images. This kind of problem is difficult to solve, because the signal is noisy, and we frequently have to use some knowledge about a specific scene. So we take interest in a restrictive class of objects, as bones or massive organs. We build a closed surface using B-splines tensor products (useful in CAD), and we propose two representations of closed surfaces : a cylinder with poles using one control grid, and a control graph where several grids may continuously merge. Both representations are useful for local or global deformations, or for topological transformations. We model the deformation process as a deflating balloon : starting with a shape surrounding the scene, we let it deflate in order to obtain successively the convex hull, the locally convex hull, and finally the external surface. The main interest of this approach consists in its capacity to find information about a complex object, such as topology, or such as the number of connected parts. The segmentation method uses this technique on a set of edge indices, detected in the images. We can see this method as a tool to model the natural interpolation effect of the human eye. This work is therefore a middle-level operator, which gives a global coherence from geometric properties of the images density function. This does not solve the problem of *a priori* knowledge, but a user interface can be used where we need some specific information.

## REMERCIEMENTS

*Je suis très reconnaissant à Jacques Demongeot qui m'a accueilli dans son laboratoire, ainsi qu'à Roger Mohr et Philippe Cinquin qui ont co-dirigé cette thèse, et dont je tiens à souligner le soutien qu'ils m'ont apporté tout au long de ce travail.*

*Je voudrais également remercier chacun des membres de mon jury de thèse de l'honneur qu'ils m'ont fait d'assister à ma soutenance, ainsi que pour la discussion fructueuse qui en a suivi. Je pense plus précisément à mes deux rapporteurs, Henri Maître et Michel Pierre, pour la patience et le soin avec lesquels ils ont examiné mon manuscrit. Je suis très reconnaissant à Claude Puech de s'être intéressé à mon travail et d'avoir fait partie de ce jury.*

*Merci à l'équipe TIMC, et plus particulièrement à la salle 106, avec Eric Bainville, Georges Bettega, Mohamed Djaïd, François Estève, Thomas Fortin, Olivier François, Pascal Sautot, Jacques Vermont, qui ont su créer une agréable ambiance de travail.*

*Enfin merci à Marian & Michel Hamelin, pour le temps qu'ils ont accordé à la constitution de mon manuscrit, et à d'autres occupations plus ludiques au cours de ma thèse.*



## TABLE DES MATIERES

CHAPITRE 1 : INTRODUCTION .....	9
1- Cadre général .....	10
2- Segmentation d'images .....	11
3- Contraintes de réalisation.....	12
4- Approche du problème .....	13
CHAPITRE 2 : ETUDE BIBLIOGRAPHIQUE .....	15
1- Introduction .....	16
2- La segmentation .....	16
2.1- Généralités .....	16
2.2- Détection des points frontière.....	17
2.2.1- Contours et régions.....	17
2.2.2- Quelques opérateurs de détection de contours.....	18
2.3- Mise en cohérence des points frontière .....	20
2.4- Conclusion.....	23
3- Les surfaces.....	23
3.1- Les tableaux de voxels.....	24
3.2- La triangulation.....	24
3.4- Les surfaces de forme libre .....	25
3.5- Conclusion.....	27
CHAPITRE 3 : REPRESENTATION DE SURFACES DEFORMABLES .....	29
1- Introduction .....	30
2- Choix de la représentation.....	30
3- Evaluation pratique d'une surface fermée .....	31
3.1- Subdivision d'un produit tensoriel de B-splines .....	31
3.1.1- Rappel : définition des B-splines .....	31
3.1.2- Rappel : subdivision d'une courbe spline.....	32
3.1.3- Evaluation d'une surface spline.....	35
3.2- Raccordement polaire.....	40
3.3- Graphe de contrôle.....	43
4- Propriété de raffinement local.....	46
4.1- Raffinement local d'une grille de contrôle.....	46
4.2- Raffinement polaire .....	49
4.3- Raffinement d'un graphe de contrôle .....	51
5- Auto-intersection d'une surface.....	52
5.1- Deux types d'intersections .....	52
5.2- Le type "trou".....	53
5.2.1- Cas d'une grille régulière .....	53
5.2.2- Cas d'un graphe de contrôle.....	56
5.3- Le type "étranglement" .....	58
5.3.1- Cas d'une grille régulière .....	58
5.3.2- Cas d'un graphe de contrôle.....	61
6- Mise en œuvre logicielle.....	62
6.1- Codage des coordonnées des points .....	62
6.2- Evaluation des surfaces.....	62
6.2.1- Evaluation d'une grille de contrôle .....	62
6.2.2- Evaluation d'un graphe de contrôle.....	64
6.3- La structure surface .....	65
6.3.1- Représentation par grilles régulières.....	65
6.3.2- Représentation par graphe .....	68
6.4- La structure scène .....	71
7- Conclusion .....	72
CHAPITRE 4 : DYNAMIQUE DE LA SURFACE.....	75
1- Introduction .....	76



1.1-Préambule.....	76
1.2- Hypothèses initiales.....	76
1.3- Méthode utilisée.....	77
2- Structures de données.....	77
2.1- Structure de données "scène à modéliser".....	77
2.2-Structure de données "surface déformable".....	80
2.3-Structure de données de contrôle.....	80
3- Contraintes internes.....	81
3.1- Stratégie de déformation.....	81
3.1.1- Utilisation de la courbure locale.....	81
3.1.2- Choix des points à déplacer.....	82
3.1.3- Grilles régulières : traitement des pôles.....	85
3.2- Contraction convexe.....	86
3.2.1- Déplacement des points elliptiques convexes.....	86
3.2.2- Adaptation de la continuité géométrique.....	87
3.3- Contraction hyperbolique.....	89
3.3.1-Elasticité directionnelle aux points hyperboliques.....	89
3.3.2-Auto intersection.....	91
3.3.3- Détection d'étranglement.....	94
3.4- Contraction concave.....	95
3.4.1- Adaptation aux concavités.....	95
3.4.2-Auto intersection.....	98
3.4.3-Détection de trou.....	99
3.5- Raffinements.....	101
3.5.1- Généralités.....	101
3.5.2- Raffinement par subdivisions successives.....	102
3.5.3- Raffinement quelconque.....	103
4- Contraintes externes.....	103
4.1- Prise en compte de l'environnement.....	103
4.2- Contrôle du déplacement.....	105
5- Conclusion.....	106
<b>CHAPITRE 5 : APPLICATION A LA SEGMENTATION.....</b>	<b>107</b>
1- Introduction.....	108
2- Données à segmenter.....	108
2.1-Types de données 3D.....	108
2.2- L'effet de volume partiel.....	110
3- Détection d'indices de contours.....	112
3.1- Méthode.....	112
3.2- Détection de lignes de crête du gradient.....	112
3.3- Construction d'éléments de surface (ou indices).....	113
4-Mise en cohérence de ces contours.....	115
4.1- Mise en place d'une surface déformable.....	115
4.2- Evolution de cette surface.....	116
4.3-Elimination d'indices de contours.....	116
4.4- Création d'indices de contours.....	117
4.5- Résultats.....	118
5- Introduction de connaissance à priori.....	121
5.1- Principe.....	121
5.2-Adaptation d'une forme prédéfinie.....	122
6- Conclusion.....	124
<b>CHAPITRE 6 : CONCLUSION.....</b>	<b>125</b>
1- Introduction.....	126
2- Représentation de surfaces.....	126
3-Déformation de surfaces.....	127
4- Segmentation.....	127
5- Applications.....	128

5.1- Sculpture virtuelle.....	128
5.2- Segmentation interactive.....	129
6- Perspectives de recherche .....	130
6.1- Segmentation .....	130
6.1.1- Exploitation d'une périodicité probable .....	130
6.1.2- Introduction de connaissance a priori .....	131
6.2- Parallélisation.....	131
REFERENCES BIBLIOGRAPHIQUES .....	133
ANNEXES.....	141

### LISTE DES ILLUSTRATIONS

Subdivision d'un polygone de contrôle d'une courbe spline fermée.....	32
Subdivision d'un polygone de contrôle d'une courbe spline ouverte .....	35
Polyèdre de contrôle de la surface spline .....	37
Une étape de subdivision sur les lignes.....	37
Polyèdre de contrôle intermédiaire après l'opération sur les lignes.....	38
Une étape de subdivision sur les colonnes du polyèdre intermédiaire .....	38
Première subdivision du polyèdre de contrôle de la surface spline .....	38
Raccordement implicite des bords du produit tensoriel .....	39
Contrainte $G^0$ pour le raccordement polaire d'une nappe cylindrique d'ordre 3.....	41
Raccordement polaire d'une surface spline cylindrique d'ordre 3.....	42
Evaluation d'un cylindre d'ordre 4 à raccordements polaires .....	43
Schéma de subdivision d'un graphe .....	44
Subdivisions successives d'un tétraèdre de contrôle.....	45
Surfaces d'ordre 2, 3, 4, 5 issues d'un tétraèdre.....	46
Raffinement au point $Q_4$ par rapport à la première subdivision de $(P_0, \dots, P_3)$ .....	47
Raffinement d'une grille de contrôle à la première subdivision.....	48
Deux raffinements successifs d'un cylindre.....	49
Construction du repère polaire $R_{p\acute{o}le}$ à partir de trois points mobiles.....	50
Déformation du repère $R_{p\acute{o}le}$ en $R'_{p\acute{o}le}$ et des points polaires.....	51
Deux raffinements successifs d'une structure à facettes .....	52
Carreaux en contact en O avant construction du trou.....	54
Réordonnancement des raccordements pour la construction d'un trou .....	54
Représentation du raccordement en trou .....	55
Raccord entre deux surfaces à deux subdivisions différentes .....	56
Création d'un trou dans une structure à facettes .....	57
Exemple d'une surface à trou d'ordre 4 .....	57
Grille de contrôle initiale.....	59
Grille de contrôle à l'étranglement.....	59
Cylindre à pôles sectionné à l'équateur à partir de la première subdivision .....	60
Double tétraèdre de contrôle sectionné à la première subdivision.....	61
Mise en œuvre de l'algorithme de subdivision.....	63
Passes de subdivision ligne et colonne pour une surface.....	64
Structure de données d'une surface déformable .....	66
Élément de surface sur-subdivisé en deux zones .....	68
Structure de données "Point" .....	69
Structure de données "Facette" .....	69
Structure de données d'une surface issue d'un graphe de contrôle.....	70
Ajout de points dans un graphe de contrôle .....	71
Structure de scène avec un objet à trou localement sursubdivisé .....	72
Algorithme synthétique de déformation d'une surface spline subdivisée $s_f$ fois.....	78
Bitmap 3D arborescente .....	79
Problème du choix de l'amplitude du déplacement d'un point.....	82
Position initiale du polygone de contrôle .....	83
Intersection avec l'objet après déplacement simultané de $P_1$ et $P_2$ .....	84

Intersection avec l'objet après recul de $P_2$ .....	84
Contraintes de déplacement au pôle .....	86
Déplacement "barycentrique" des points de contrôle elliptiques convexes .....	87
Adaptation à une forme convexe.....	88
Position avant l'étape de progression tangentielle .....	89
Position finale des points $P_1$ et $P_2$ sur l'angle droit .....	89
Choix des directions de déplacement pour un point de contrôle hyperbolique.....	90
Adaptation à une forme présentant une selle .....	92
Adaptation à une forme "double" .....	93
Structure d'auto-test.....	94
Répulsion d'un point par ses voisins .....	96
Adaptation à une forme présentant des concavités.....	97
Direction de déplacement des concavités selon la normale rentrante .....	98
Adaptation à une forme torique .....	100
Progression dichotomique d'un point du polyèdre.....	105
Balayage en mode projectif.....	109
Mode à coupes parallèles .....	109
Mode à coupes non parallèles.....	110
Objet test pour une acquisition parallèle .....	110
Résultat discrétisé de l'acquisition.....	111
Utilisation d'un contour paramétré sur des données discrètes .....	111
Cas de contour multiple et cas de contour trop épais (en 2D).....	114
Contraction de la surface fermée initiale sur un examen de vertèbre .....	119
Contraction de la surface fermée initiale sur un examen de rein .....	120
Repositionnement spatial de coupes échographiques de reins .....	121
Une transformation élastique d'un modèle vers un objet .....	123
Interface de sculpture virtuelle .....	129

# **CHAPITRE 1**

## **Introduction**

## 1- Cadre général

Le domaine dans lequel se situe ce travail est celui de la segmentation d'images, ou de manière plus générale, celui de l'interprétation d'images. Plus précisément, on utilise ici des images de type médical, qui représentent une variété assez importante de problèmes particuliers.

De manière générale, la segmentation représente une étape importante dans le traitement d'images, puisqu'elle se situe à la charnière entre l'acquisition des images et l'utilisation de la sémantique qu'elles contiennent.

En effet, toutes les applications pratiques automatisées respectent une démarche en trois étapes, définies par une acquisition de données, un traitement, et une réponse en fonction du type de l'application. Le traitement des données revient à les traduire dans un formalisme tel que le système en question puisse effectuer la tâche qui lui est assignée. Quand les données fournies sont des images, il faut traduire ces images en des entités géométriques telles que des contours (ou de manière complémentaire des zones). C'est cette traduction que l'on appelle l'étape de segmentation.

Dans une chaîne de traitement d'images médicales, on retrouvera ce besoin de traiter l'information brute des images, de manière à dégager des différents types d'examen une sémantique en terme d'anatomie. Le résultat de la segmentation des images sera alors utilisé pour le diagnostic, la quantification de zones ou le repérage d'un outil.

A cause de la grande diversité des organes humains, tant au niveau de la composition qu'au niveau du fonctionnement, on utilise différents modes d'acquisition (radiographie, tomographie, angiographie, résonance magnétique nucléaire, échographie, Doppler...), et l'on dispose donc d'une grande variété de types d'images à traiter. Chacun de ces modes a ses imperfections à prendre en compte, et les formes à détecter sont des plus diverses en termes de dimensions. Certaines correspondent à des objets ayant un volume, qui peut parfois avoir une topologie complexe, soit naturellement, soit par suite d'un processus pathologique. On rencontre ensuite des objets surfaciques (membranes) et des objets filaires (vaisseaux). Enfin le mode d'acquisition peut engendrer un effet de texture sur les données : on repère alors des motifs plutôt que des zones homogènes. Cette variété de formes induit une même variété de problèmes de segmentation, et il semblerait que chacun de ces problèmes doive être résolu séparément, dans un premier temps du moins. En effet, il n'existe pas encore de solution générale à la détection des contours.

Notons que ces différents modes d'acquisition fournissent de plus un signal parfois fort bruité, et que les données numériques qui en sont issues sont loin d'être optimales : le rapport signal sur bruit est parfois décevant, comme sur certaines angiographies.

## 2- Segmentation d'images

Il apparaît que le problème est difficile à résoudre notamment à cause de la piètre qualité des signaux acquis, et de la nécessité, souvent incontournable, d'avoir une connaissance spécifique pour l'interprétation d'une image. Remarquons au passage que le "bruit" associé au signal des images, est justement qualifié de "bruit" puisque'il ne correspond souvent pas à ce que l'observateur voit ou veut voir. Le problème de l'élimination du bruit est donc intimement lié à l'injection de connaissance dans le processus de détection.

La littérature [HAR] est d'ailleurs remplie de méthodes fonctionnant "bien" sur de "bonnes" images. Elles ont cependant l'immense importance de préparer le terrain à un deuxième étage de méthodes qui devraient élargir la classe des "bonnes" images. En effet, on s'est longtemps contenté d'utiliser strictement les informations contenues dans les pixels des images alors que l'observateur "complète" l'image, inconsciemment ou non. Il faut noter que ces "ajouts" peuvent être classés selon :

- des critères physiologiques (capacité de l'oeil à voir des lignes continues).
- le contexte (surprise visuelle, changement brutal de sujet).
- la connaissance générale de l'observateur (différencier un motif d'un objet).
- la connaissance spécifique de la scène acquise (l'oeil du spécialiste).

Ceci montre le fossé qui sépare l'observateur humain des machines munies d'un programme de segmentation : ces dernières ne repèrent que des pixels, sont éternellement surprises par les images qu'on leur fournit, et n'ont à peu près aucune connaissance, qu'elle soit générale ou spécifique.

Certains auteurs [ZUC][MAR][CAN] ont analysé les images à l'aide des discontinuités de la fonction densité associée, et repèrent donc des "indices de contours". Pour relier ces indices dans un tout cohérent, sont nées des méthodes de suivi de contours [LIU][CAP][MON]. Elles restent cependant des opérateurs trop locaux (bien qu'il existe des solutions multi-échelles), et récemment, on a commencé à voir apparaître des méthodes dites de contours actifs [TER][COH][ROU]. En fait ces méthodes représentent un outil de modélisation de l'effet d'interpolation et de lissage de l'œil humain, lié à une représentation continue des contours. De bons résultats ont vu le jour et c'est dans cette gamme d'outils que nous proposons une solution plus adaptée aux données tridimensionnelles.

### 3. Contraintes de réalisation

Il semble illusoire d'espérer trouver une méthode tout à fait générale qui résoudrait le problème de la segmentation. Il faut cependant dès à présent prévoir un système large, prenant en compte tous les niveaux du problème, en partant du préfiltrage de données jusqu'à l'intégration de connaissance a priori.

Un tel système peut donc être conçu comme un logiciel entièrement contrôlé par l'utilisateur, dans lequel on va introduire des procédures automatiques, résolvant tel ou tel problème. Le travail qui suit correspond donc à la réalisation de ces procédures.

On va s'intéresser à une restriction au cas des objets de type volumique qui sont les plus courants, tels que des os ou des organes massifs. Les méthodes de contours actifs utilisent un contour déformable soumis à un système de forces. Ces forces sont telles que la résolution (de diverses manières) de l'équation de la dynamique associée donne le contour de l'objet étudié. L'extension tridimensionnelle de cette méthode implique la modélisation d'une surface déformable, qui va chercher à rejoindre les indices de contours d'une forme. On s'attachera à développer une représentation capable de modéliser toutes les formes à traiter, tout en vérifiant un certain nombre de caractéristiques essentielles :

- rapidité d'évaluation. En effet, l'évolution au cours du temps de la surface demande un nombre important d'évaluations (c'est à dire de calcul des coordonnées) des points qui la constituent.
- aptitude à subir des transformations topologiques. C'est impératif pour pouvoir s'adapter à tous les objets à traiter.
- portabilité du format (nécessaire pour le passage dans des logiciels de visualisation ou de CAO).
- facilité de calcul de caractéristiques géométriques de l'objet, très utilisées en chirurgie, telles que l'aire ou le volume.

Compte tenu des critères énoncés ci-dessus, on utilisera une modélisation consistant en une fonction interpolante qui va épouser les formes recherchées via un maillage de points. De plus, cette forme de modélisation permettra par la suite d'archiver les résultats sous forme de base de données d'objets, dans une représentation compatible avec les logiciels de CAO, et permettra plus tard de les réutiliser sous forme d'une connaissance *a priori*.

Par ailleurs, et pour reprendre les principes généraux du début de ce chapitre, les résultats d'un traitement sont codés selon un certain formalisme, que l'on pourra traduire à son tour dans le formalisme de la chaîne de réponse. Cela peut-être une simple image, des

instructions à une machine, ou encore les cotes d'un objet. Dans notre cas, une méthode à base de surfaces sera donc pratique d'utilisation pour le rendu d'une scène, et surtout ses résultats pourront être appliqués au plan médical : planning chirurgical, éventuellement préparation de prothèses.

#### **4- Approche du problème**

Pour atteindre les objectifs précédemment exposés, on va adopter dans les chapitres suivants une démarche en plusieurs étapes :

- dans un premier temps, le chapitre deux exposera une étude bibliographique succincte dans le but de positionner ce travail par rapport aux travaux existants, que ce soit vis-à-vis des méthodes de segmentation d'images, ou des différents types de représentation de surfaces.

- dans un deuxième temps on prévoit la construction d'un outil d'interpolation de données tridimensionnelles par le biais d'une surface déformable. La caractéristique essentielle des données à traiter est qu'elles définissent un objet dont la topologie est au départ inconnue. Pour cela, on procède en deux sous-étapes :

- \* dans le chapitre trois, on choisit une représentation de surfaces adaptée aux déformations à l'aide de fonctions splines, que l'on détaillera, de l'algorithme d'évaluation des surfaces splines jusqu'à la gestion d'une transformation topologique. On résoudra en particulier le problème de gestion des pôles pour une forme topologiquement équivalente à une sphère, et de la hiérarchie des raffinements continus des points de contrôle des carreaux de B-splines. On distinguera deux types de transformation topologiques, de la sphère en tore ("le trou") et de la sphère en deux sphères ("l'étranglement"), qui résument les possibilités de changement que pourra subir la surface déformable.

- \* la gestion au cours du temps d'une telle surface, décrite dans le chapitre quatre, concerne la manière de choisir le déplacement de ses points de contrôle. On verra que c'est la topologie de la forme à rejoindre qui va régler le déplacement de ces points et que l'on peut définir un classement naturel des formes en fonction des changements de topologie de cette surface.

- après avoir préparé cet outil, on l'appliquera à la segmentation proprement dite dans le chapitre cinq. Les données à traiter seront les indices de contours des données réelles, détectés à l'aide d'opérateurs classiques. L'outil fournira alors une cohérence globale sur les éléments ainsi détectés. On traitera en particulier le problème de



l'élimination de "faux" indices de contours, et à l'opposé la construction de "nouveaux" indices, non détectés par des opérateurs de bas niveau.

- enfin on concluera sur l'utilisation d'un tel outil dans le cadre de la mise en cohérence d'éléments de contours, et surtout sur ses perspectives de développement et d'utilisation.

## **CHAPITRE 2**

### **Etude bibliographique**

## 1 Introduction

Cette étude a pour but de situer le travail présenté ici, et de justifier en partie certains choix intervenant dans les chapitres suivants.

Le problème de la segmentation sera exposé dans le premier paragraphe (2.2). On verra rapidement diverses méthodes de détection d'éléments "frontière" des images, et la mise en cohérence de ces éléments pour définir les objets décrits par l'image. On survolera alors le problème (incontournable) d'introduction de connaissance *a priori*.

On fera, dans le second paragraphe (2.3), un rapide tour d'horizon des différentes manières de représenter une surface dans l'espace, et qui puissent s'appliquer à la reconstruction tridimensionnelle des frontières d'un objet.

## 2- La segmentation

### 2.1- Généralités

Le problème de la segmentation d'image est apparu avec la nécessité d'obtenir les contours de tel ou tel objet dans une image, pour des raisons diverses. Le terme de segmentation s'est alors lié au problème puisque l'on cherche à découper l'image à l'aide des contours des objets. En fait, il apparaît que la définition la plus reconnue du problème consiste à partitionner l'image (généralement une matrice carrée de points), en autant de zones que l'observateur estime qu'il y a d'objets représentés dans cette image. Notons que dans certaines images, des zones physiquement disjointes peuvent appartenir au même objet (un objet peut être partiellement masqué). On définit le terme de sur et sous-segmentation par le fait que le nombre de zones détectées est supérieur ou inférieur au nombre d'objets que l'on voit dans l'image.

On va cependant voir que cette définition fort simple se révèle ardue à prendre en compte, dès que les images contiennent des informations d'une richesse élevée, tant au point de vue de la sémantique de l'image, que des nuances des informations elles-mêmes (certaines images ont une information codée sur 2048 niveaux de gris, alors que l'œil humain ne perçoit bien que 64 niveaux). Il est en effet important de noter que ce problème de segmentation fait intervenir d'une part de manière évidente les informations contenues dans la matrice de points, mais aussi les informations liées intrinsèquement à l'image, que l'on appelle connaissance *a priori*. Pour faciliter la résolution du problème, on peut donc le découper en deux sous-problèmes (non forcément disjointes) : la détection et l'interprétation :

Typiquement, les images en niveaux de gris obtenues en photographie ou selon des modalités plus complexes (systèmes d'imagerie médicale), apportent des problèmes de détection de contours, dûs à la présence de bruit de fond. De plus, les scènes représentées sont parfois complexes, et une étape d'interprétation de ces contours se révèle indispensable.

Un exemple d'images où le problème d'interprétation apparaît seul, est celui des images "synthétiques" (en général binaires), telles que des pages d'écriture, ou certains dessins. En effet, le propre du dessin, ou de l'écriture, est d'être une représentation purement sémantique, qui ne pose que peu de problèmes de détection : de la concision du dessin dépend la clarté du message. L'image correspondante peut alors être facilement séparée en zones (par des techniques classiques), mais leur cohérence est difficile à exprimer si l'on ne connaît pas le type de sémantique utilisée.

On a jusque là parlé de l'interprétation de la scène, en rapport avec la connaissance que l'on a du sujet. Mais considérons maintenant des images de faible difficulté sémantique. Il apparaît alors que la vision humaine contient des automatismes dont le plus important (en terme de détection) est la faculté d'interpoler des données. Par exemple une succession de points alignés, mais non connexes, sera vue comme une droite, et de même pour des arcs de courbes. Des contours difficiles à détecter, car ne contenant que peu de points "sûrs", seront quand même bien vus par l'œil grâce à ce mécanisme. Beaucoup d'images se trouvent dans ce cas, soit directement, soit après un traitement de base. En effet, le bruit provenant de l'acquisition des données tend souvent à fournir des détails "en pointillés", invisibles au premier coup d'œil, mais fort gênants pour un programme à base d'opérateurs locaux.

Aussi peut-on essayer d'appréhender le problème de la segmentation par une découpe du problème en trois sous-problèmes :

- la détection des caractéristiques (points frontières) réellement existantes dans l'image,
- la mise en œuvre de mécanismes de base pour rendre cohérents ces points frontières,
- enfin, l'intégration de la connaissance "humaine" de la scène.

## **2.2- Détection des points frontière**

### **2.2.1- Contours et régions**

Dans le cadre de la recherche des contours appartenant à une image, bon nombre de

méthodes ont vu le jour [HAR], et se caractérisent en deux grandes classes :

- les approches par détection de contours proprement dites, que nous détaillons plus loin.

- les approches par régions regroupant des pixels faisant partie d'un même objet. En fait, cette approche est concurrente de la première, mais rencontre le même genre de problèmes quant au bruit et à la discrétisation. Nous ne nous étendrons pas sur cette catégorie de détecteurs, peu en rapport avec la méthode présentée ici. Sommairement, elle consiste en des algorithmes d'étiquetage, qui partitionnent l'image en zones présentant des propriétés d'homogénéité (variation de l'intensité lumineuse par exemple). On peut les subdiviser en 2 catégories :

- \* les méthodes de classification : on détermine d'abord une partition de niveaux de gris (par calcul d'histogramme), puis on utilise les relations spatiales existant entre les pixels pour former des régions.

- \* les méthodes de croissance de régions : on regroupe itérativement des points ou ensemble de points qui ont des propriétés de connexité, en régions plus importantes suivant des critères d'homogénéité.

Compte tenu de la lourdeur de mise en œuvre dans le cas 3D, on adopte nécessairement un compromis : soit le critère de fusion est relativement complexe, mais on réalise en fait une série de détections 2D, soit le processus d'aggrégation se fait sur les trois axes, et le critère de fusion est simplifié [RHO].

### 2.2.2- Quelques opérateurs de détection de contours

Comme les bords d'un objet sur une image sont caractérisés par des discontinuités de la fonction densité, on peut les détecter suivant deux approches classiques :

- calcul du gradient et extraction des maxima locaux de la norme du gradient dans sa direction (ou recherche de zéros de la dérivée du gradient dans sa direction).

- calcul du laplacien et recherche des points où il s'annule (zero-crossing).

Des études théoriques ont été menées [HAR] sur le problème de la différenciation dans le cas 2D, et ont conduit à des approches rigoureuses, soit générales, soit aboutissant à un détecteur précis, par une analyse mathématique.

Le choix et l'importance du filtrage lissant préliminaire est un point essentiel, mais l'efficacité du détecteur dépend principalement de l'étape de différenciation. Ainsi, il faut tenir compte des points suivants : une recherche de passage par zéro est plus facile à faire qu'une recherche d'extremum (il suffit de tester deux valeurs, alors qu'un extremum est désigné par son voisinage). Il en est de même de l'utilisation d'un opérateur isotrope

plutôt que des opérateurs directionnels. Par contre les recherches de zéros peuvent induire de fausses détections (zones "trop plates"). De plus les points frontières détectés par le laplacien et la dérivée du gradient dans sa direction peuvent ne pas coïncider : en effet, les filtres utilisés sont des approximations discrètes des fonctions gradient et laplacien.

En 3D, les mêmes problèmes se posent et les généralisations de gradient se trouvent sous la forme de filtres simples, conduisant à des masques de convolution. Certains auteurs ont mis au point des opérateurs optimaux en définissant théoriquement les performances d'un bon détecteur, suivant le modèle de discontinuité et la quantité de bruit sur les données. En particulier, les opérateurs de Marr [MAR] et Canny [CAN] ont été généralisés en 3D [WIT][MON1]. Bien entendu, ces méthodes donnent des indices de contours non nécessairement connectés entre eux (il reste à la charge d'un autre opérateur de les relier).

On peut citer quelques filtres optimaux désormais classiques :

- l'opérateur de Zucker et Hummel [ZUC] : un point frontière est considéré comme un plan passant par le centre d'une sphère. On définit alors ce plan de manière optimale. C'est une généralisation 3D de l'opérateur de Sobel.

- l'opérateur de Marr et Hildreth [MAR] : c'est un opérateur appelé aussi chapeau mexicain, qui est basé sur un pré-filtrage à base de gaussienne, suivi d'un opérateur laplacien. La variance de la gaussienne sera estimée en fonction de la taille des informations à détecter : ceci ouvre alors la voie à des méthodes "multi échelles" [WIT].

- l'opérateur de Canny [CAN] est basé sur l'optimisation de la localisation, la stabilité, et l'unicité de détection des points frontières. On convolue d'abord à l'aide d'une gaussienne, puis on utilise un opérateur directionnel. Cet opérateur améliore la détection et la localisation des contours, par rapport au laplacien.

Ces opérateurs de filtrage ont donné le jour à une suite, appelée "opérateurs de diffusion". Le point de départ est le modèle de Marr et Hildreth [MAR] généralisé par Witkin [WIT]. On convolue le signal par des gaussiennes de variances croissantes et on définit à chaque échelle des points frontières comme des maxima du gradient ainsi régularisé. Witkin a montré que cela revenait à résoudre l'équation de la chaleur avec le signal comme condition initiale. L'inconvénient majeur de cette méthode est la mauvaise localisation des points frontières, due au phénomène de diffusion. Perona et Malik [PER] proposent de diffuser le signal de manière sélective, uniquement aux points de gradient faibles (les points frontières étant caractérisés par des gradients forts) selon un seuil à déterminer. Le principal problème inhérent à ce critère est qu'il suppose que les

fluctuations dues au bruit ont un gradient plus faible que celui des points frontières. Ce type de méthodes fait depuis l'objet de recherches [CAT].

Toutefois, il apparaît que telle qu'est construite l'équation d'évolution en fonction du temps, le système ne passe jamais par une position d'équilibre stable. A la longue toutes les inhomogénéités disparaissent, et il faut impérativement une action de l'utilisateur pour contrôler ce processus.

De manière générale, la seule utilisation de ces opérateurs donne des résultats limités à cause du fait que l'œil perçoit, par interpolation ou extrapolation, des éléments non contenus dans les données de l'image. Ce phénomène est d'autant plus marqué que le signal est bruité, ce qui limite l'efficacité des opérateurs locaux, les versions des opérateurs étendues à une vaste zone restant lourdes et peu précises. Des méthodes multi-échelle seront donc plus adaptées puisque travaillant des plus gros éléments vers les plus petits.

### **2.3- Mise en cohérence des points frontière**

Le principal problème de toutes ces méthodes de recherche de discontinuités est de relier les points frontières entre eux pour former un véritable contour (ou surface), d'éliminer ceux qui correspondent au bruit et de remplir les trous éventuels. Dans le cas d'images bien contrastées, l'opérateur utilisé peut être suffisant pour obtenir effectivement un ensemble de points frontières entièrement connectés. En réalité, le bruit lié aux modalités d'imagerie fait que les profils des discontinuités sont rarement idéaux et bien plus souvent estompés.

Un certain nombre de techniques issues du suivi de contour dans le cas 2D ont été généralisées en 3D pour relier les "vrais" points frontières et éliminer les autres. La phase de correction de l'étape d'extraction des points frontières peut utiliser des techniques interactives ou automatiques :

- Liu [LIU] propose un algorithme de suivi de surface dans lequel il insère une phase de correction d'erreur. Chaque nouveau point frontière est testé selon le critère du gradient maximum mais aussi par rapport à un seuil adaptatif.

- Cappelletti et Rosenfeld [CAP] ont élaboré une fonction de coût permettant de suivre les contours 2D dans chaque coupe mais en utilisant l'information 3D : elle est basée sur la moyenne du gradient 3D de Zucker et Hummel [ZUC] sur un ensemble de points frontières, la "circularité" du point candidat et un critère de proximité avec les contours des coupes adjacentes. Le suivi de contour se fait par un algorithme de recherche de graphe optimisant la fonction de coût.

Il existe encore bon nombre de méthodes envisagées pour connecter les indices de contours [MON2], mais dans l'ensemble, ces techniques restent trop locales pour espérer regrouper tous ces indices dans un ensemble cohérent, de manière sensiblement automatique. En effet, elles ont souvent recours à des paramètres de réglage dont le choix de la valeur est toujours un casse-tête, généralement résolu de manière interactive.

Dans le cadre de la reconstruction de surfaces, assez proche du suivi de surfaces, citons les "marching cubes" de Lorensen [LOR], qui construisent des modèles triangulaires de surfaces d'isodensité. L'algorithme utilise pour cela une table d'intersections de contours pour décrire comment la surface coupe chaque cube élémentaire d'un volume de données.

Dans le modèle du système visuel, on découvre un outil d'interpolation entre les détails d'une image. On remarque que les méthodes de segmentation s'orientent petit à petit vers une prise en compte des formes à détecter. Dans le cas des formes lisses une manière simple de les interpoler consiste à partir d'un contour déjà existant, et de le déformer sur des indices de contours. Cette technique, introduite par Kass, Witkin et Terzopoulos [KAS] [TER] a été reprise par de nombreux auteurs [BER] [COH] [ROU], et a reçu la dénomination générale de "snakes", par analogie à un serpent qui se love sur les données, la version française étant les "contours actifs". Après avoir choisi une courbe englobant l'objet recherché, on la laisse évoluer en la soumettant à des forces internes (représentant l'élasticité et la régularité de la courbe), et des forces externes (représentant les points frontière à rejoindre). Le modèle se déforme donc sous l'action de ces forces jusqu'à atteindre une position d'équilibre représentant un minimum du système de forces appliqué, correspondant en principe au contour désiré. Formellement, on cherche à minimiser l'énergie de la courbe définie par :

$$E = E_{int} + E_{image} [+ E_{ext}]$$

On définit un contour déformable par  $v$  :

$$\Omega = [0,1] \rightarrow \text{Image}$$

$$s \rightarrow v(s) = (x(s), y(s))$$

-  $E_{int}$  représente l'énergie interne de la courbe et impose la régularité et la rigidité du modèle :

$$v \rightarrow E_{int}(v) = \int_{\Omega} (a \|v'(s)\|^2 + b \|v''(s)\|^2) ds \quad \text{où } v' \text{ et } v'' \text{ sont les dérivées de } v$$

Si  $a = 0$ , on retrouve le critère de minimisation remplie par les fonctions splines d'interpolation (bien que l'on ne puisse parler d'interpolation, le but du modèle étant justement de rechercher les points du contour).

-  $E_{image}$  tend à amener la courbe vers les points frontières selon un critère de minimisation du potentiel  $P(x,y)$  :



$$E_{\text{image}} = \int_{\Omega} P(v) \, ds \text{ avec par exemple } P(x,y) = -\|\overrightarrow{\text{grad}I(x,y)}\|^2$$

où  $I(x,y)$  représente l'intensité lumineuse de l'image au point  $(x,y)$ .

-  $E_{\text{ext}}$  impose les contraintes externes définies par l'utilisateur (par exemple un point fixe).

On minimise alors l'énergie totale en résolvant l'équation d'évolution associée, par une méthode de différences finies.

Un certain nombre d'améliorations ont été proposées pour résoudre des problèmes inhérents à la méthode originale de Terzopoulos [KAS]:

- au lieu de minimiser le périmètre du contour, on peut minimiser l'aire intérieure du contour pour détecter plus facilement les structures non convexes [FRN].

- pour assurer une meilleure convergence sur les points frontière détectés dans l'étape de filtrage, on peut simuler un potentiel à partir des ces données (à l'aide d'une convolution par une gaussienne) pour attirer le contour actif [COH] [LEI1] [GUE].

- la résolution de l'équation d'évolution se fait en discrétisant la courbe en un certain nombre de points sur lesquels on évalue le champ de forces. Deux solutions ont été proposées pour faire intervenir un bilan des forces à partir de chaque segment de courbe compris entre deux points de la subdivision initiale de la courbe. L'intérêt est de diminuer le nombre de points dans la discrétisation de la courbe et de rendre les résultats plus stables, du fait de "l'intégration" du champ de force le long du segment de courbe.

\* on peut modéliser la courbe initiale par des fonctions B-splines, et intégrer le champ de forces en chaque point du polygone de contrôle représentant la courbe. L'équation d'évolution de la courbe est alors établie sur les points de contrôle. L'intérêt consiste en la manipulation d'un petit nombre de points [LEI1] [GUE]. Ce type d'approche correspond en fait à la solution qui est présentée dans ce travail.

\* la solution préconisée par Cohen reste plus proche du modèle initial, puisqu'elle consiste à utiliser une technique d'éléments finis pour la résolution de l'équation d'évolution plutôt que des différences finies.

- l'initialisation reste la plupart du temps un problème. En effet, si l'on cherche à rejoindre un objet (par l'extérieur ou l'intérieur), des structures étrangères peuvent être présentes. Il est alors difficile de trouver une position de départ, autrement qu'interactivement. Notons une idée intéressante de Rougon [ROU], qui semble satisfaisante : il utilise des "marqueurs déformables", construits à partir de morpho-

mathématiques, qui donnent une première approximation des contours d'objets. Celle-ci convient pour initialiser le contour actif.

## 2.4- Conclusion

Il apparaît que le problème de la segmentation reste non résolu, de manière générale du moins.

On peut cependant estimer qu'en termes d'opérateurs de bas niveau, les auteurs ont su apporter toute une gamme de produits directement utilisables, dont certains sont très performants. La recherche en ce domaine n'est cependant pas terminée.

Le domaine des opérateurs de niveau intermédiaires, développés pour relier le résultat des précédents dans un tout cohérent, commence à se remplir de méthodes à base de snakes, qui obtiennent de bons résultats. C'est d'ailleurs dans ce domaine que nous proposons d'inclure le travail présenté ici.

Par contre, on rencontre encore peu de choses sur une méthodologie de l'introduction de connaissance *a priori*. C'est certainement dans cette direction qu'auront lieu les prochains apports pour la segmentation, en la spécialisant à un domaine.

## 3- Les surfaces

L'intérêt de modéliser une surface est de pouvoir faire des traitements multiples et répétés comme la visualisation, le zoom, la quantification par des paramètres morphologiques (volume, aire), ou l'application à des outils de CAO (pour la construction de prothèses par exemple). Des méthodes d'approximation de surfaces (tableaux de voxels, triangulation, carreaux de Bézier ou de B-splines) ont été développées initialement pour de objets "technologiques", puis pour la synthèse d'images. On se référera à la synthèse de Böhm [BOH] sur toutes les techniques de modélisation de surface, et à l'étude comparative de Barsky [BAR] concernant les modèles 3D à base de surfaces de forme libre.

Présentons donc succinctement dans les paragraphes suivants, par ordre croissant de degré de continuité, trois grands types de représentation de surface. On s'attachera à référencer les travaux axés sur la reconstruction tridimensionnelle d'objets, à partir de données portant sur leurs contours. En effet, nous verrons dans les chapitres suivants, que l'utilisation que nous ferons des surfaces rentre dans ce contexte.

### 3.1- Les tableaux de voxels

L'acquisition des données constituant la surface se fait de manière numérique, et le produit de cette numérisation consiste en une matrice de voxels. La dimension de ces voxels est bien sûr en relation avec les qualités de finesse de l'appareil d'acquisition.

Dans ces conditions, la représentation la plus naturelle d'une surface consiste à chaîner dans une liste l'ensemble des voxels constituant la surface [LIU] [GOR] [UDU].

La surface de l'objet est alors décrite comme un ensemble de voxels. On parle de 0-voxel et de 1-voxel correspondant respectivement au fond et à l'objet. La surface est définie comme un graphe dirigé, où les nœuds sont les centres des faces d'un 1-voxel frontière (c'est à dire qu'au moins une face ou une arête est commune à un 0-voxel), et les liens sont des arêtes.

L'intérêt essentiel de cette représentation réside dans le fait qu'elle est très simple, et que les algorithmes associés le sont aussi. On peut cependant lui faire deux critiques :

- bien qu'elle soit exactement représentative de la discrétisation de la surface correspondante, il est difficile d'estimer le caractère "lisse" de l'objet ainsi modélisé.
- lors d'une déformation éventuelle de cette surface, il sera difficile de conserver le même nombre de voxels : il faudra créer, ou supprimer, des listes de voxels dans la structure de graphe.

### 3.2- La triangulation

On cherche à reconstruire la surface d'un objet en interpolant ses contours par des facettes triangulaires. Différents critères d'interpolation sont utilisés :

- maximiser le volume du polyèdre [KEP 75] composé par tous les triangles dans le cas d'un objet concave. Dans le cas d'un objet convexe, on cherchera plutôt à minimiser ce volume.
- minimiser la surface du polyèdre [FUC 88] formé par les triangles.

Dans chacune de ces approches, le problème de la triangulation se ramène à une recherche de plus court chemin dans un graphe. Des améliorations ont été apportées pour traiter le cas des embranchements et des trous dans les objets, par ajout de singularités.

Boissonat [BOI] a proposé d'utiliser la triangulation de Delaunay pour apparier les points de contours des objets. Citons aussi Edelsbrunner [EDE] pour ses  $\alpha$ -shapes, qui obtient

des résultats impressionnants sur des objets de topologie complexe. Les  $\alpha$ -shapes sont une généralisation de l'enveloppe convexe d'un ensemble de points  $S$  : une  $\alpha$ -shape est caractérisée par le complémentaire du volume dont chacun des points est le centre d'une sphère de rayon  $\alpha$  ne contenant aucun point de  $S$ .

Des algorithmes spécifiques de visualisation 3D permettent d'obtenir des images lissées des objets ainsi modélisés. L'intérêt de ces méthodes est de représenter l'information de façon concise et facilement manipulable. On peut ainsi calculer facilement des paramètres quantitatifs comme le volume, et même étudier les propriétés mécaniques des objets, en utilisant cette représentation sur des stations de CAO.

Cependant, cette technique peut être critiquée sur deux points :

- l'approximation polygonale des objets peut supprimer des irrégularités et des détails qui peuvent se révéler très importants pour l'objet étudié. En fait, pour éviter ce problème, il faudrait que chaque voxel du contour de l'objet soit un sommet du polygone : on aurait alors une représentation beaucoup plus précise. Le nombre de facettes devenant considérable, les temps de calcul augmentent et diminuent alors l'intérêt de la méthode.

- malgré les algorithmes de lissage de surfaces composées de facettes (employés pour la visualisation), les surfaces obtenues restent de moins bonne qualité que celles obtenues à partir de surfaces de forme libre.

### **3.4- Les surfaces de forme libre**

Les modélisations de surfaces par tableaux de voxels ou par triangulation ne fournissant pas des approximations suffisamment lisses pour certaines applications, des représentations à base de carreaux polynomiaux ont été développées.

Pour des applications de segmentation d'images médicales, la première approche a été proposée par Sequeira [SEQ], qui utilise des primitives de forme libre à base de B-splines : une primitive est choisie et placée sur l'objet étudié. Notons qu'il faut ré-échantillonner ce modèle pour obtenir une distribution uniforme des points de contrôle des B-splines. En se plaçant dans des cas où la surface peut être représentée en coordonnées cylindriques ou sphériques, on peut simplifier le problème. On utilise alors avec de très bonnes performances des carreaux de B-splines [CHA] ou de Bézier [DU].

Pour formuler l'équation (voir la synthèse de [FAR]) de telles surfaces, on utilise les polynômes de Bernstein (carreaux de Bézier), ou les B-splines de base. Ces surfaces sont définies en utilisant des points de contrôle, et il existe par ailleurs une formulation rationnelle, qui fait intervenir des poids en plus des points de contrôle. Les NURBs (Non

Uniform Rational B-splines) [PIE] utilisent une répartition non uniforme des noeuds des fonctions de base. La modification de la valeur d'un des poids n'entraîne alors pas une modification de toute la surface.

Un problème rencontré par ces surfaces, obtenues à partir d'un produit tensoriel de fonctions polynomiales, est celui des formes de topologie arbitraire. De manière générale, le problème revient à créer des carreaux à  $n$  côtés, alors que le produit tensoriel n'en donne naturellement que quatre. Bon nombre de solutions ont vu le jour, et on peut citer Sabin [SAN], qui utilise des carreaux sans produit tensoriel, limités à cinq côtés. Loop et DeRose [LOO] proposent des généralisations des surfaces B-splines biquadratiques et bicubiques, en utilisant une généralisation de carreaux de Bézier à  $n$  côtés. Citons encore Du [DU] pour la construction de surfaces à partir de facettes triangulaires de Bernstein-Bézier.

Ouvrons une parenthèse à propos de l'intersection de surfaces, dont nous aurons besoin par la suite. Dans le cas d'intersection de surfaces polynomiales, plusieurs solutions sont envisageables (synthèse de [PAT]) [AZI] :

- les méthodes analytiques, qui posent des problèmes quant au degré, et à la puissance de calcul requise.

- les méthodes travaillant sur les intersections de courbes isoparamétriques, qui réduisent la complexité du problème, mais qui peuvent omettre des solutions.

- les méthodes générant des séquences de points en explorant la courbe "intersection" dans une direction définie par une étude locale de la surface. Le point de départ est généralement trouvé en utilisant la méthode précédente. Des problèmes peuvent provenir du "pas d'avance" de l'exploration de la courbe, en menant à des boucles sans fin ou à des connexions erronées.

- les méthodes par subdivision, qui décomposent récursivement le problème de l'intersection en des problèmes plus simples, jusqu'à des intersections triviales (ex plan/plan). Il suffit ensuite de connecter les solutions individuelles. Cette démarche n'a donc pas besoin de point de départ, mais peut se révéler lente si on recherche une grande précision.

Il existe donc un certain nombre de solutions pour la détection d'intersection de surfaces polynomiales, et les méthodes réalisant une étude locale discrète semblent présenter le meilleur coût efficacité/temps de calcul.

De manière générale, il apparaît que l'utilisation de surfaces de forme libre donne de très bons résultats en matière d'approximation et de reconstruction. Notons cependant qu'elles sont assez lourdes à mettre en œuvre dans le cas général.

### 3.5- Conclusion

Le but de la modélisation de la surface des objets est double : c'est bien entendu un moyen d'approximer une surface, cohérente au sens de la sémantique de l'objet, en fonction de l'ensemble des voxels constituant ses frontières. C'est de plus un moyen de compression de données, permettant de prendre en compte la morphologie propre à l'objet.

Les deux premiers types de représentation de surface, sous forme de tableaux de voxels et de triangles, ont donné lieu à des algorithmes de construction très performants, parfois commercialisés. Elles ont cependant un inconvénient commun : la représentation par facettes comprime sous forme polygonale les objets ayant des surfaces lisses, ce qui ne correspond parfois pas du tout à ce que l'on recherche.

En fait les systèmes de CAO ou de visualisation (Euclide, A.V.S.), fournissent maintenant systématiquement la possibilité d'utiliser des surfaces lisses (sous forme de carreaux polynomiaux). Les formes libres sont certainement plus lourdes à mettre en œuvre, mais jouissent d'une meilleure adaptation aux différents objets possibles. En effet, elles permettent de retrouver des aspects de continuité de certains objets, qui ont été perdus par discrétisation, lors de leur numérisation.



## **CHAPITRE 3**

### **Représentation de surfaces déformables**



## 1- Introduction

Dans ce chapitre, nous cherchons à réaliser une surface active, qui sera utilisée dans le cadre de la segmentation (chapitres 4 et 5). A la fin de la procédure de segmentation, cette surface active devra définir complètement l'objet à segmenter. Aussi allons-nous choisir une représentation de surface d'une part adaptée aux déformations, et d'autre part adaptable aux objets à segmenter.

Dans un premier temps (paragraphe 2), on définira deux modes de représentation susceptibles de convenir à l'énoncé ci-dessus.

Dans un deuxième temps (paragraphe 3), on détaillera ces représentations, et principalement la méthode d'évaluation du lieu géométrique des points de la surface.

Puis, on donnera de meilleures caractéristiques de déformation à cette surface en lui conférant des propriétés de raffinement local (paragraphe 4).

Enfin (paragraphe 5), on verra comment traiter d'éventuelles transformations topologiques (de la sphère au tore, et d'une sphère en deux sphères).

Le paragraphe 6 montrera les choix de programmation retenus pour ces deux représentations.

## 2- Choix de la représentation

L'utilisation de la surface déformable consiste à représenter progressivement des surfaces gauches d'objets quelconques, qui peuvent avoir des degrés de continuité élevés. L'utilisation principale de l'outil développé dans ce travail se faisant dans un cadre médical, beaucoup d'objets à représenter se révèlent avoir des contours lisses. Une surface déformable ayant par nature une continuité élevée ( $C^2$  par exemple) serait la bienvenue. Aussi préférera-t-on une surface de type polynomial (dont il suffit de régler le degré en fonction du degré de continuité voulu) à une représentation sous forme de liste de points ou de facettes.

Un objet quelconque peut cependant avoir des discontinuités : il faut alors choisir une représentation polynomiale par morceaux et décrire ces discontinuités au moyen de raccords adaptés, d'où la notion de carreaux de surface.

Deux solutions sont possibles pour gérer les carreaux de surfaces :

- soit considérer une liste de carreaux munis de leurs voisins, sans ordre particulier,
- soit utiliser une grille de carreaux, munie de conditions aux bords.

La liste de carreaux est particulièrement souple d'emploi, notamment dans le cas des surfaces fermées, mais implique une duplication des voisinages, pénalisante à

l'évaluation quand le nombre de carreaux se multiplie (par exemple, lorsqu'on augmente la souplesse en un endroit). A l'opposé la solution en grille fournit un ordre sur la surface, et se trouve particulièrement adaptée pour l'évaluation de la surface, puisqu'elle peut se définir sous forme d'un produit tensoriel. C'est pourquoi on penche dans un premier temps pour cette dernière solution. On verra cependant aussi une solution à base de carreaux, issue de l'algorithme d'évaluation.

Deux formes de représentation apparaissent alors, à savoir la forme fonctionnelle ( $f(x,y,z) = 0$ ) et la forme paramétrique ( $P = f(u,v)$ ). Là encore c'est le type d'application qui va trancher :

- la première forme s'adapte bien à l'écriture de prédicats de type *appartient*(P), où P est un point de l'espace, et à l'évaluation d'un point particulier du graphe de la fonction.

- la seconde forme s'adapte à l'évaluation directe de la surface par balayage de l'espace paramétrique, mais est plus lourde pour évaluer le prédicat précédent, bien que cette solution permette d'élaborer un échantillonnage précalculé de la surface, utile si la surface ne se modifie pas et qu'un grand nombre d'évaluations de ce prédicat est à faire. Cependant, les problèmes d'évolution d'une surface dans l'espace étant plus liés à de multiples évaluations du lieu géométrique de ses points qu'à un test ponctuel d'appartenance à celle-ci, on s'oriente vers une représentation paramétrique d'une surface polynomiale.

La représentation d'une telle surface va être réalisée au moyen d'un produit tensoriel de fonctions splines pour une raison de simplicité d'évaluation. Les fonctions splines utilisées sont les B-splines de base d'un degré adapté à l'objet - en pratique des cubiques - sur des noeuds entiers.

### **3- Evaluation pratique d'une surface fermée**

#### **3.1- Subdivision d'un produit tensoriel de B-splines**

##### **3.1.1- Rappel : définition des B-splines**

Soit  $\tau = \{t_i, i \in I\}$  une suite strictement croissante d'abscisses, appelée noeuds : I est un ensemble d'indices qui peut être fini. On définit [BOO] [MEH] les B-splines d'ordre 1 :

$$N_i^1(x) = 1 \text{ si } t_i \leq x < t_{i+1}, 0 \text{ sinon}$$

et les B-splines d'ordre  $k > 1$  par la relation de récurrence suivante :

$$N_i^k(x) = \frac{x - t_i}{t_{i+k-1} - t_i} N_i^{k-1}(x) + \frac{t_{i+k} - x}{t_{i+k} - t_{i+1}} N_{i+1}^{k-1}(x)$$

Etant donnés des points  $\{P_i, i \in I\}$  de l'espace ( $\mathbb{R}^3$  par exemple), une courbe spline  $S_k$  d'ordre  $k$  de points de contrôle  $P_i$  est une combinaison linéaire des points  $P_i$ , dont les coefficients sont des B-splines d'ordre  $k$  :

$$(1) \quad S_k(x) = \sum_{i \in I} P_i N_i^k(x) \quad \text{où } x \in J = \left( \inf_i t_i, \sup_i t_i \right)$$

On désigne par  $N_n(x)$  la B-spline d'ordre  $n$  et de noeuds  $0, 1, \dots, n$ . On suppose que  $\tau = Z$  (noeuds entiers). En effet, on va par la suite augmenter le cardinal de  $t$  en le multipliant par  $p$ , et il sera plus simple de travailler avec des noeuds du type  $j/p, p \in Z$ .

En utilisant cette notation, on définit  $S$  par :

$$S(x) = \sum_{j \in Z} P_j N_n(x - j) \quad \text{sachant que } \sum_{j \in Z} N_n(x - j) = 1$$

### 3.1.2- Rappel : subdivision d'une courbe spline

La méthode de subdivision [MIC] [SAB] est basée sur la représentation d'une courbe spline sur une suite de noeuds plus fine :  $\tau_p = \{j/p, j \in Z\}$  où  $p$  est un entier  $> 1$ .

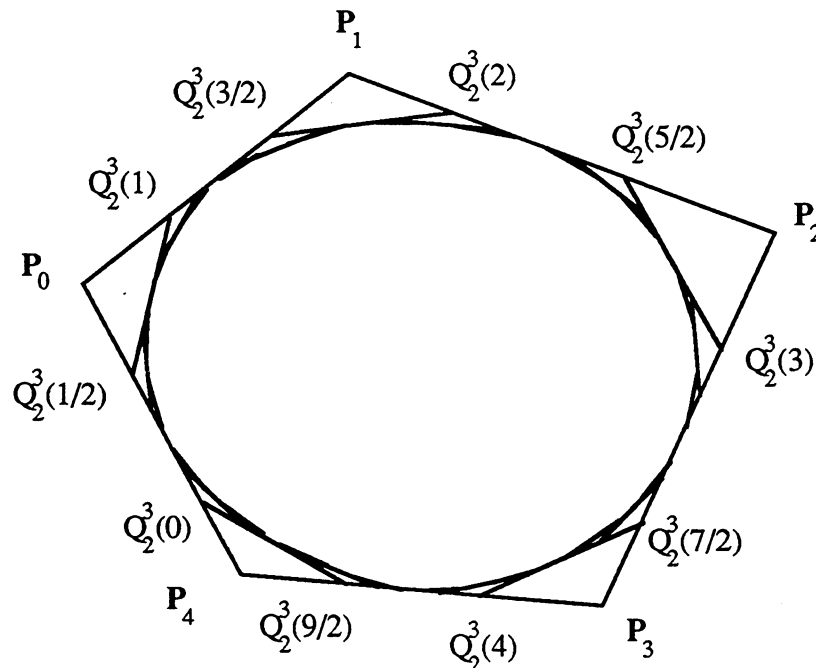


Figure 1 : subdivision d'un polygone de contrôle d'une courbe spline fermée

On introduit les constantes  $\alpha_p^n(\frac{j}{p})$  par  $N_n(x) = \sum_{j=0}^{n(p-1)} \alpha_p^n(\frac{j}{p}) N_n(px - j)$

où les nouvelles B-splines  $N_n(px-j)$  ont leurs noeuds aux points de  $Z + 1/p$  avec  $0 \leq j \leq p-1$ .

On obtient alors  $S(x) = \sum_{k \in Z} Q_p^n(\frac{k}{p}) N_n(px - k)$  où  $Q_p^n(\frac{k}{p}) = \sum_{i \in Z} \alpha_p^n(\frac{k-i}{p-1}) P_i$

On démontre au moyen des transformées de Fourier des B-splines, que les coefficients  $\alpha_p^n(\frac{j}{p})$  sont

ceux du développement  $\frac{1}{p^{n-1}} \left( \frac{z^p - 1}{z - 1} \right)^n = \sum_{j=0}^{n(p-1)} \alpha_p^n(\frac{j}{p}) z^j$

En multipliant cette dernière équation par  $\frac{1}{p} \left( \frac{z^p - 1}{z - 1} \right) = \frac{1}{p} (1 + z + \dots + z^{p-1})$

on obtient la relation  $\alpha_p^{n+1}(\frac{j}{p}) = \frac{1}{p} \sum_{i=0}^{p-1} \alpha_p^n(\frac{j-i}{p-1})$

et en remarquant que  $Q_p^1(j) = \dots = Q_p^1(j + \frac{p-1}{p}) = P_j \quad (j \in Z)$

il vient la relation de récurrence suivante :

$$(2) \quad Q_p^{n+1}\left(\frac{k}{p}\right) = \frac{1}{p} \sum_{i=0}^{p-1} Q_p^n\left(\frac{k-i}{p}\right) \quad (k \in Z)$$

Il faut donc, après avoir augmenté le nombre de points de contrôle (passage de la suite  $\tau$  à la suite  $\tau_p$ ),  $p$  applications de cette relation de récurrence pour obtenir une subdivision plus fine du polygone de contrôle. Partant du polygone périodique  $(P_0, \dots, P_4)$  précédent (figure 1), supposé être le polygone de contrôle fermé d'une courbe spline d'ordre 3 ( $n$  est égal à 2), on obtient en 2 étapes le polygone associé à une subdivision 2 fois plus fine.

Pour  $p$  égal à 2, (2) devient  $Q_2^{n+1}\left(\frac{k}{2}\right) = \frac{1}{2} \left( Q_2^n\left(\frac{k}{2}\right) + Q_2^n\left(\frac{k-1}{2}\right) \right) \quad (k \in Z/(2 \times 5)Z)$

L'étape initiale fournit donc par dédoublement les points suivants :

$$Q_2^1(0) = Q_2^1\left(\frac{1}{2}\right) = P_0, \quad Q_2^1(1) = Q_2^1\left(\frac{3}{2}\right) = P_1, \quad Q_2^1(2) = Q_2^1\left(\frac{5}{2}\right) = P_2$$

$$Q_2^1(3) = Q_2^1\left(\frac{7}{2}\right) = P_3, \quad Q_2^1(4) = Q_2^1\left(\frac{9}{2}\right) = P_4$$

On applique une première fois la récurrence :

$$Q_2^2(0) = \frac{1}{2}(P_4 + P_0), Q_2^2\left(\frac{1}{2}\right) = P_0, Q_2^2(1) = \frac{1}{2}(P_0 + P_1), Q_2^2\left(\frac{3}{2}\right) = P_1, Q_2^2(2) = \frac{1}{2}(P_2 + P_1), Q_2^2\left(\frac{5}{2}\right) = P_2$$

$$Q_2^2(3) = \frac{1}{2}(P_2 + P_3), Q_2^2\left(\frac{7}{2}\right) = P_3, Q_2^2(4) = \frac{1}{2}(P_3 + P_4), Q_2^2\left(\frac{9}{2}\right) = P_4,$$

Le second niveau de récurrence fournit alors les positions des noeuds de la première subdivision :

$$Q_2^3(0) = \frac{1}{4}P_0 + \frac{3}{4}P_4, Q_2^3\left(\frac{1}{2}\right) = \frac{3}{4}P_0 + \frac{1}{4}P_4, Q_2^3(1) = \frac{1}{4}P_1 + \frac{3}{4}P_0, Q_2^3\left(\frac{3}{2}\right) = \frac{3}{4}P_1 + \frac{1}{4}P_0$$

$$Q_2^3(2) = \frac{1}{4}P_2 + \frac{3}{4}P_1, Q_2^3\left(\frac{5}{2}\right) = \frac{3}{4}P_2 + \frac{1}{4}P_1, Q_2^3(3) = \frac{1}{4}P_3 + \frac{3}{4}P_2, Q_2^3\left(\frac{7}{2}\right) = \frac{3}{4}P_3 + \frac{1}{4}P_2$$

$$Q_2^3(4) = \frac{1}{4}P_4 + \frac{3}{4}P_3, Q_2^3\left(\frac{9}{2}\right) = \frac{3}{4}P_4 + \frac{1}{4}P_3$$

A partir des  $P_i$ , on peut donc calculer les  $Q_2^3(i)$ , et obtenir ainsi par subdivisions successives la courbe spline correspondante (figure 1).

L'évaluation d'une courbe spline ouverte fonctionne sur le même principe. Seule la construction de la suite  $\tau_p$  diffère : dans le cas d'une courbe fermée, on duplique  $p$  fois chaque noeud de la suite  $\tau$ . Ici on ne duplique pas les points extrêmes de  $\tau$ , et il n'y a pas de périodicité.

Partant d'un polygône ( $P_0, \dots, P_3$ ), supposé être le polygone de contrôle ouvert d'une courbe spline d'ordre 3, on obtient de nouveau en 2 étapes le polygone associé à une subdivision 2 fois plus fine.

$$(1) \text{ devient } Q_2^{n+1}\left(\frac{k}{2}\right) = \frac{1}{2}\left(Q_2^n\left(\frac{k}{2}\right) + Q_2^n\left(\frac{k}{2} - \frac{1}{2}\right)\right) \text{ si } \inf(\tau_2) \leq k \leq \sup(\tau_2)$$

$$Q_2^{n+1}\left(\frac{k}{2}\right) = \inf(\tau_2) \text{ si } k < \inf(\tau_2)$$

$$Q_2^{n+1}\left(\frac{k}{2}\right) = \sup(\tau_2) \text{ si } k > \sup(\tau_2)$$

L'étape initiale fournit donc par dédoublement les points suivants :

$$Q_2^1(1) = Q_2^1\left(\frac{3}{2}\right) = P_1, Q_2^1(2) = Q_2^1\left(\frac{5}{2}\right) = P_2$$

On applique une fois la récurrence :

$$Q_2^2(1) = \frac{1}{2}(P_0 + P_1), Q_2^2\left(\frac{3}{2}\right) = P_1, Q_2^2(2) = \frac{1}{2}(P_2 + P_1)$$

$$Q_2^2\left(\frac{5}{2}\right) = P_2, Q_2^2(3) = \frac{1}{2}(P_2 + P_3)$$

Le second niveau de récurrence fournit alors les positions des noeuds de la première

subdivision :

$$Q_2^3(1) = \frac{1}{4}P_1 + \frac{3}{4}P_0, \quad Q_2^3\left(\frac{3}{2}\right) = \frac{3}{4}P_1 + \frac{1}{4}P_0, \quad Q_2^3(2) = \frac{1}{4}P_2 + \frac{3}{4}P_1$$

$$Q_2^3\left(\frac{5}{2}\right) = \frac{3}{4}P_2 + \frac{1}{4}P_1, \quad Q_2^3(3) = \frac{1}{4}P_3 + \frac{3}{4}P_2, \quad Q_2^3\left(\frac{7}{2}\right) = \frac{3}{4}P_3 + \frac{1}{4}P_2$$

Aussi de même que pour une courbe fermée, on est en mesure d'évaluer la courbe spline en reportant les  $Q_2^3(i)$  subdivision par subdivision (figure 2).

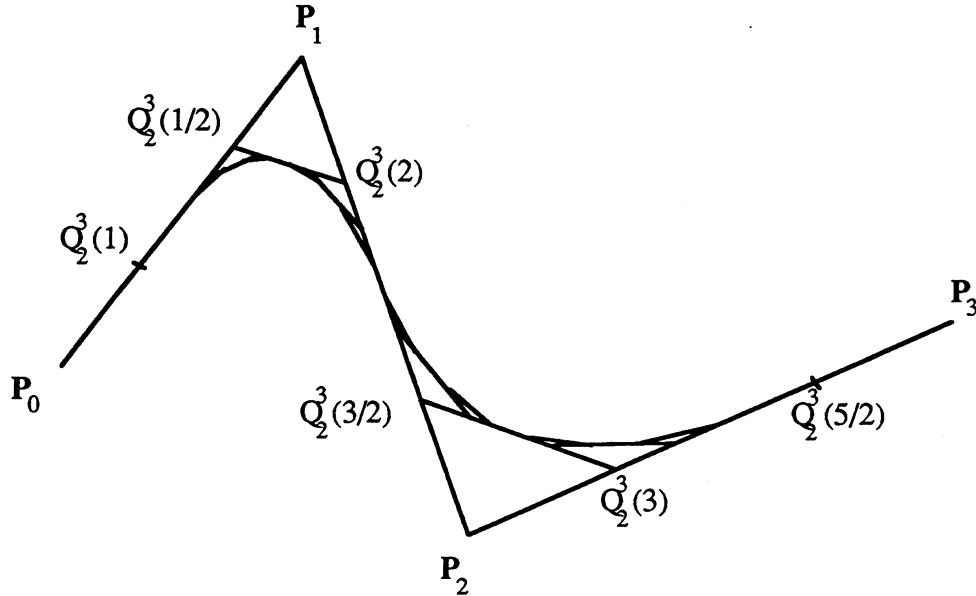


Figure 2 : subdivision d'un polygone de contrôle d'une courbe spline ouverte

### 3.1.3- Evaluation d'une surface spline

Sachant évaluer les points d'une courbe spline au moyen d'un algorithme de subdivision, on définit une surface par le produit tensoriel de deux fonctions splines (chacune associée à une suite de noeuds). On utilisera un algorithme de subdivision pour la surface, directement inspiré du précédent.

En reprenant les notations ci-dessus une surface (ou nappe paramétrée) s'écrit :

$$(3) \quad S_k(x,y) = \sum_{i \in I} \sum_{j \in J} P_{ij} N_i^k(x) N_j^k(y)$$

$$\text{où } x \in K = \left( \inf_i u_i, \sup_i u_i \right) \text{ et } y \in L = \left( \inf_i v_i, \sup_i v_i \right)$$

En utilisant les résultats du paragraphe précédent, on va généraliser la technique précédente. Posons :

$$S(x,y) = \sum_{(i,j) \in Z^2} P_{ij} N_n(x-j) N_m(y-j)$$

où  $N_n(x)$  est la B-spline de d'ordre  $n$  et de noeuds  $0, 1, \dots, n$  (respectivement  $N_m(y)$ ). Si l'on pose :

$$N_n(x) = \sum_{i=0}^{n(p-1)} \alpha_p^n \left(\frac{i}{p}\right) N_n(px - i)$$

On obtient alors :

$$S(x,y) = \sum_{(k,l) \in Z^2} Q_p^{nm} \left(\frac{k}{p}, \frac{l}{p}\right) N_n(px - k) N_m(py - l)$$

$$\text{où } Q_p^{nm} \left(\frac{k}{p}, \frac{l}{p}\right) = \sum_{(i,j) \in Z^2} \alpha_p^n \left(\frac{k}{p} - i\right) \alpha_p^m \left(\frac{l}{p} - j\right) P_{ij}$$

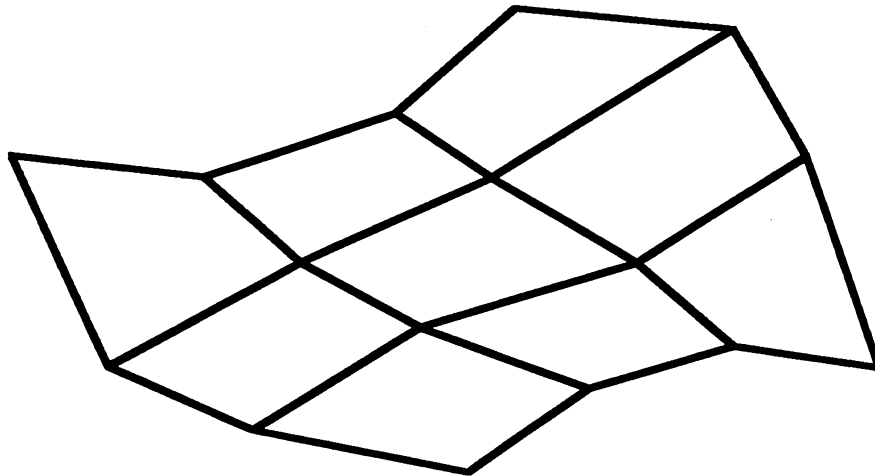
Compte tenu de la séparabilité des produits tensoriels de polynômes, on obtient une formule de récurrence équivalente à celle vue précédemment :

$$(4) \quad Q_p^{n+1 \ m+1} \left(\frac{k}{p}, \frac{l}{p}\right) = \frac{1}{p^2} \sum_{i=0}^{p-1} \sum_{j=0}^{p-1} Q_p^{nm} \left(\frac{k-i}{p}, \frac{l-j}{p}\right) \quad (k, l \in Z)$$

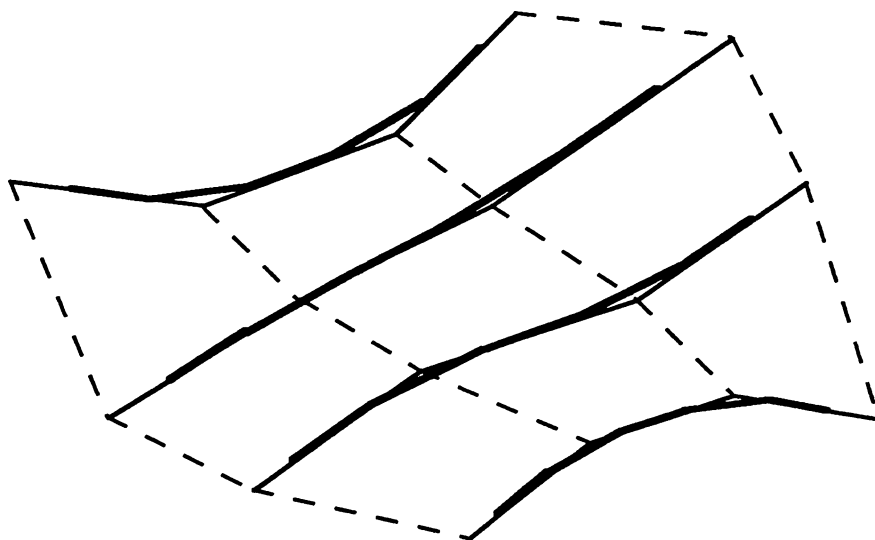
Remarquons que l'équation (4) réalise le calcul des barycentres des points (affectés d'un coefficient égal à 1) de toutes les sous-grilles de côté  $p$  contenues dans la grille de contrôle.

On peut utiliser la même méthode de construction d'une surface que pour une courbe spline. Ici, on dispose de  $m$  suites de  $n$  noeuds, que l'on va rendre plus deux fois plus fines à chaque subdivision.

La séquence suivante de schémas illustre l'algorithme : à partir du polyèdre initial, on duplique les deux suites de noeuds (figure 3). Puis on applique l'algorithme de subdivision sur les noeuds "ligne" (figure 4), ce qui nous donne un nouveau polyèdre, dupliqué en ligne (figure 5). On réapplique le même algorithme sur les noeuds "colonne" (figure 6), et on obtient la première subdivision du polyèdre de contrôle de la surface spline initiale.



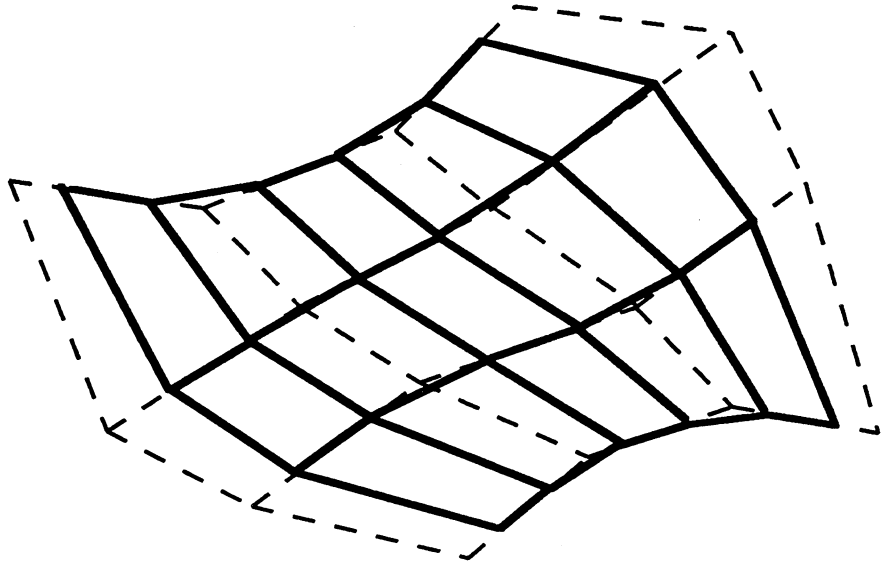
*Figure 3 : polyèdre de contrôle de la surface spline*



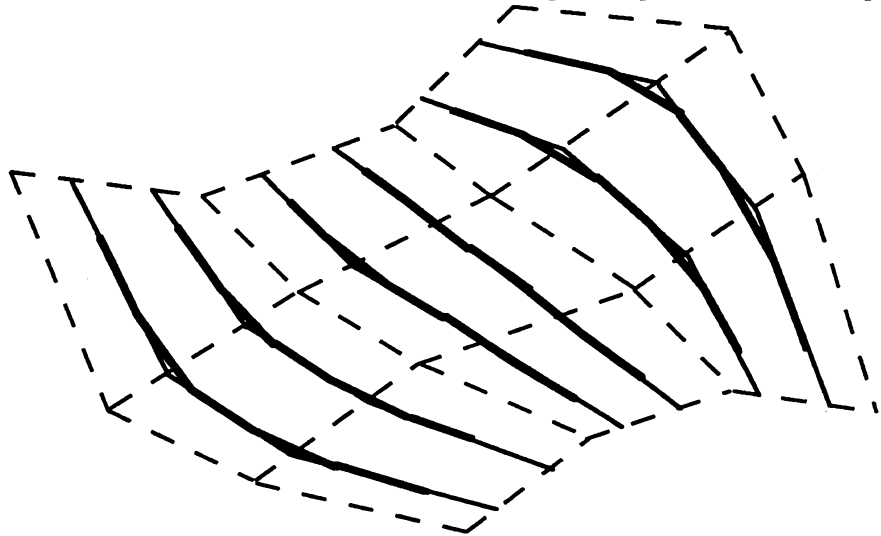
*Figure 4 : une étape de subdivision sur les lignes*

On peut montrer [MEH] que la suite des polyèdres engendrés par subdivision converge vers la surface spline initiale. Il suffit alors d'appliquer  $n$  fois cet algorithme de subdivision pour évaluer une solution approchée de la surface spline. A cause de la propriété de séparabilité, on peut procéder autrement : subdiviser  $n$  fois le polyèdre de contrôle en ligne, puis  $n$  fois en colonne. Cependant, on aura besoin par la suite de travailler sur chaque subdivision du polyèdre, et on utilisera la méthode détaillée ici.

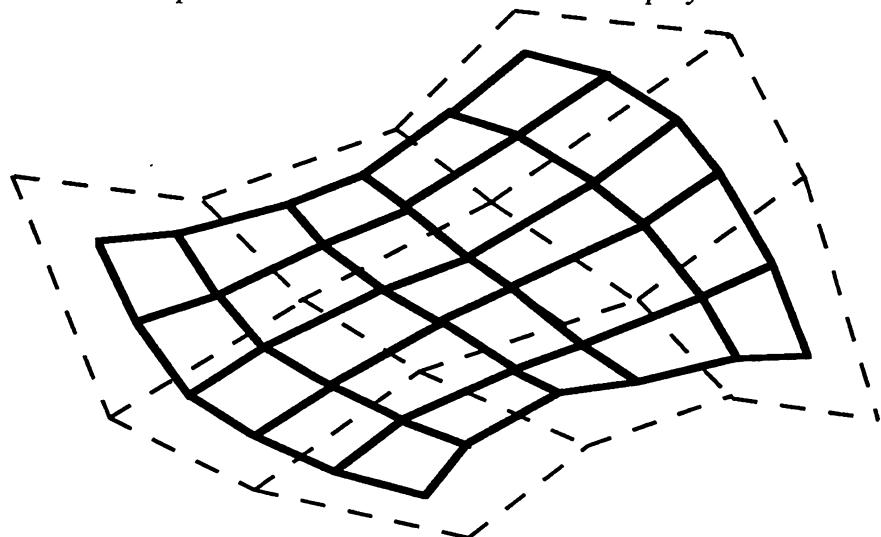




*Figure 5 : polyèdre de contrôle intermédiaire après l'opération sur les lignes*



*Figure 6 : une étape de subdivision sur les colonnes du polyèdre intermédiaire*

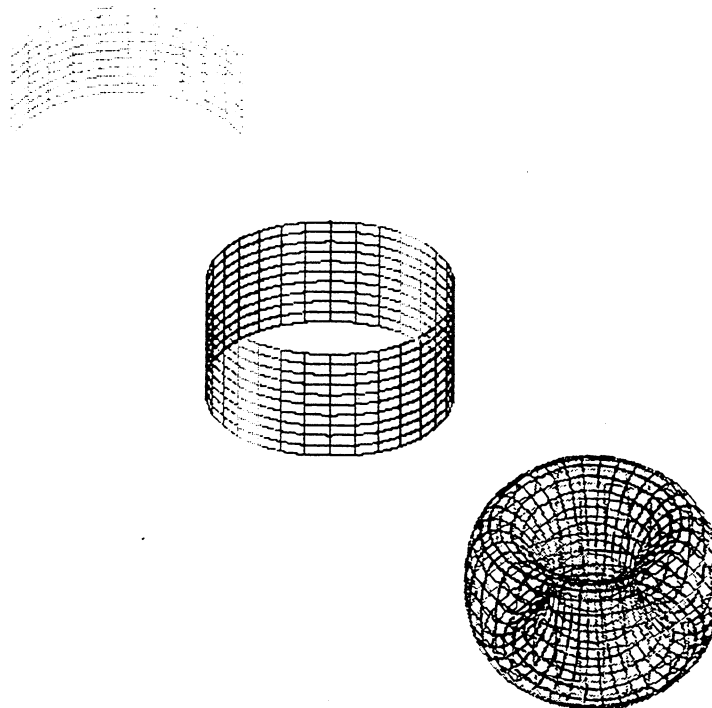


*Figure 7 : première subdivision du polyèdre de contrôle de la surface spline*

La construction de surfaces de degré quelconque sera donc faite à partir de B-splines. L'algorithme de subdivision fournit une solution approchée pour l'évaluation de telles surfaces. Les différents raccordements des bords d'une grille de contrôle conduisent à trois types de surfaces (Figure 8) :

- Le type "plan" quand les quatre bords ne sont pas raccordés.
- Le type "cylindrique" quand les bord Est et Ouest (respectivement Nord et Sud) sont raccordés deux à deux, et les bords Nord et Sud sont ouverts (respectivement Est et Ouest).
- Le type "torique" quand les bords Est et Ouest ainsi que les bords Nord et Sud sont raccordés deux à deux.

Les raccordements sont ici implicites, puisqu'ils correspondent uniquement au choix approprié de la version fermée ou ouverte de l'algorithme de subdivision. Un raccordement explicite entre deux produits tensoriels peut être généré en dupliquant suffisamment de points de contrôle sur les bords des deux polyèdres. Cela dépend bien sûr du degré de la B-spline utilisée, et cette duplication de points de contrôle s'effectue à concurrence du nombre de points du support de la B-spline.



*Figure 8 : raccordement implicite des bords du produit tensoriel*

On s'intéresse cependant à des surfaces fermées, représentant les contours d'objets tridimensionnels. Seul le type torique est implicitement fermé, mais l'initialisation d'une

telle surface présuppose l'existence d'une topologie torique sur les objets à représenter et la connaissance de la localisation du trou du tore utilisé. Aussi, un type de représentation cylindrique muni de raccordements explicites en pôles sur les bords Nord et Sud (le raccord Est-Ouest est alors implicite), permettra de construire une surface de topologie sphérique.

### 3.2- Raccordement polaire

Dans ce paragraphe, on va montrer comment réaliser un raccordement polaire c'est-à-dire comment raccorder le bord d'une grille de contrôle sur lui même, de façon à conserver de "bonnes" propriétés de continuité. De manière plus générale, on cherche à trouver des conditions simples pour raccorder en un point que l'on appellera pôle une liste de carreaux de B-spline dégénérés en ce point.

Plusieurs possibilités de fermeture au pôle peuvent être utilisées, mais la solution retenue doit satisfaire les deux conditions suivantes :

- (i) elle doit conserver une structure de produit tensoriel facilement évaluable par l'algorithme de subdivision.

- (ii) elle doit assurer une continuité de la surface au pôle à un moindre coût en termes de calcul : on verra plus tard qu'un des points clés de la méthode de déformation consiste en une évaluation répétitive des points de la surface. Compte tenu de l'adaptabilité requise pour cette surface, elle doit demeurer fermée quelles que soient les déformations imposées.

Dans la suite on raisonne sur le pôle Nord (noté PN), mais le cas du pôle Sud est identique.

De par l'utilisation systématique de grilles de contrôle, on satisfait toujours la contrainte (i).

Compte tenu de la nature cylindrique de cette surface (paragraphe 2.1.3), le bord Nord n'a aucune raison de rester dégénéré en un point au gré des déformations. On pourrait se décharger de cette responsabilité sur l'étape d'évolution de la surface, mais il est plus pratique de faire vérifier (ii) par construction, de la manière suivante.

On cherche une condition pour que la surface soit au moins  $G^0$  au pôle. Supposons la spline définie par  $n \times m$  points de contrôle  $\{P_{ij}, i = 0, \dots, n-1; j = 0, \dots, m-1\}$  sur une grille  $[0, n-1] \times [0, m-1]$ , et soit  $p$  l'ordre de la spline. Complétons la grille de points de contrôle initiale par  $p-2$  lignes identiques :

$$\{P_{ij} = P_{0j}, i = 2-p, \dots, -1; j = 0, \dots, m-1\}$$

Les courbes splines, définies par les colonnes de la grille, ont donc  $p-1$  points dupliqués

en début d'intervalle, et elles passent par ces points multiples.

Imposons  $P_{0j} = PN$  (Pôle Nord),  $j = 0, \dots, m-1$ . On vérifie sans peine, en revenant à la formule d'évaluation (3), que le choix d'une telle grille de contrôle impose

$$M(t,u) = PN, \forall u \in [0, m-1], t = 1-q$$

La figure 9 montre la réalisation d'un tel choix pour une nappe d'ordre 3 :

Dans ce cas, on a  $PN = P_{.10} = P_{.11} = P_{.12} = P_{00} = P_{01} = P_{02}$

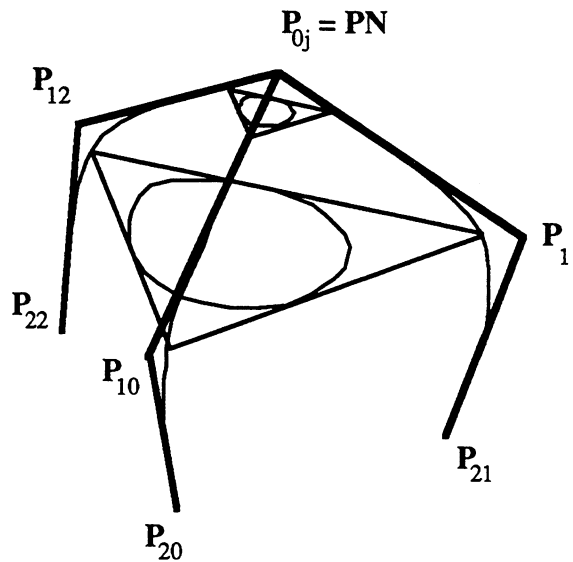


Figure 9 : contrainte  $G^0$  pour le raccordement polaire d'une nappe cylindrique d'ordre 3

On veut de plus trouver un critère pour que la surface soit  $G^1$  au pôle. Dans ce cas, la tangente au pôle dans toutes les directions doit appartenir à un même plan. Aussi, il suffit que les  $(P_{0j})$  et les  $(P_{1j})$ ,  $j \in [0, m]$ , soient coplanaires. Malheureusement, ceci rend la surface tout à fait plate au pôle. On modifie alors le contenu des lignes rajoutées :

On conserve les  $(P_{0j})$  initiaux, mais on va adapter en fonction les  $(P_{ij})$ ,  $i < 0$ . La condition de coplanarité s'applique alors sur les  $(P_{ij})$ ,  $i \leq 0$ .

\* si  $p$  est impair, considérons les courbes splines associées aux colonnes de la grille, complétées par  $p-2$  points identiques en début d'intervalle. Au voisinage du bord Nord, ces courbes se comportent comme si elle étaient d'ordre 3 (la duplication de points de contrôle fait chuter le degré de continuité géométrique de la spline utilisée). En revenant à la formule d'évaluation des courbes (2), on montre que les extrémités d'une B-spline d'ordre 3 sont les milieux des segments "extérieurs" de son polygône de contrôle. Aussi, l'expression du bord Nord est défini par la spline de polygône de contrôle :

$$\left( \frac{P_{0j} + P_{-1j}}{2} \right), j \in [0, m]$$

La fermeture du cylindre implique que cette spline soit ponctuelle au pôle PN. Aussi, on

choisit les points de contrôle des deux premières lignes non dégénérées de la façon suivante :

$$PN = \frac{1}{m} \sum_{j=0}^m P_{0j}, P_{-1j} = 2PN - P_{0j} \text{ et } P_{-ij} = P_{-1j}, \text{ avec } \forall j \in [0, m-1], \forall i > 1$$

\* si p est pair, on complète par p-3 points identiques les courbes splines associées aux colonnes de la grille. Au voisinage du bord Nord, ces courbes se comportent comme si elle étaient d'ordre 4. Soit  $(Q_i), i \in [0, n]$ , le polygône de contrôle d'une B-spline d'ordre 4. En revenant à la formule d'évaluation des courbes (2), on remarque que les extrémités de la courbe associée sont les points  $Q_1$  et  $Q_{n-1}$ , si et seulement si les triplets  $(Q_0, Q_1, Q_2)$  d'une part, et  $(Q_{n-2}, Q_{n-1}, Q_n)$  d'autre part, sont alignés. L'expression du bord Nord est alors défini par la spline de polygône de contrôle  $(P_{-1j}), j \in [0, m]$ . On choisit donc les  $(P_{ij}), i < 0$ , de façon à fermer le cylindre en PN :

$$P_{-1j} = PN = \frac{1}{m} \sum_{j=0}^m P_{0j}, P_{-2j} = 2PN - P_{0j} \text{ et } P_{-ij} = P_{-2j}, \forall j \in [0, m], \forall i < 2$$

Il faut noter que le choix des  $P_{0j}$  reste libre. On peut chercher à obtenir dans le cas  $p > 3$  une continuité  $G^2$  au pôle, ce qui implique que la courbure au pôle est la même dans toutes les directions. Ceci amène à déterminer une relation supplémentaire sur les  $P_{0j}$  et les  $P_{1j}$ . Pour tout j, les points  $P_{-1j}, P_{0j}$  et  $P_{1j}$  doivent avoir même position relative. Cette contrainte est cependant trop lourde pour être imposée aux points de contrôle lors de leurs déplacements. De plus on verra que la "bande Nord" de carreaux de B-spline sera spatialement très réduite, et une hypothèse de continuité  $G^2$  semble superflue.

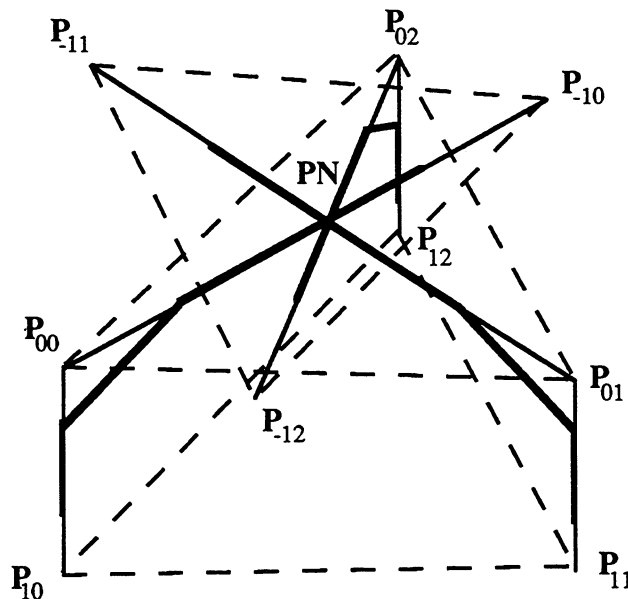
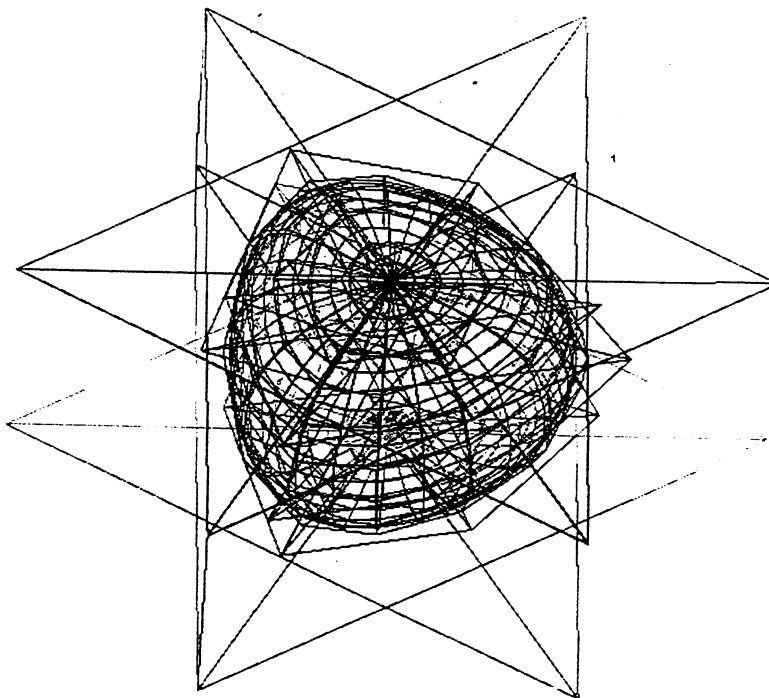


Figure 10: raccordement polaire d'une surface spline cylindrique d'ordre 3

La condition de coplanarité induit cependant une lourdeur à gérer au cours du déplacement des points. Aussi supprime-t-on ce problème en associant à la surface de

topologie sphérique un polyèdre de contrôle ne comportant que 3 colonnes, garantissant ainsi à tout moment une continuité  $G^1$  au pôle.

La figure 10 met ainsi en évidence la technique utilisée pour un cylindre à pôle d'ordre 3, sur la grille  $(P_{ij})$ ,  $(i,j) \in [0,2] \times [0,3]$ . Les subdivisions suivantes feront converger ce polyèdre vers l'hémisphère de pôle PN. La figure 11 représente la suite des subdivisions pour un cylindre à pôles d'ordre 4, partant d'une grille  $6 \times 3$ . Notons que dans ce cas il n'y a que six points de contrôle effectivement mobiles, les positions des autres étant déduites des six premiers.



*Figure 11: évaluation d'un cylindre d'ordre 4 à raccordements polaires*

### 3.3- Graphe de contrôle

Dans le paragraphe précédent, on a vu comment spécialiser certains points d'une grille de contrôle régulière, pour fermer à l'aide de pôles une surface de type cylindrique. On peut se demander s'il n'est pas possible de trouver une forme plus générale de représentation, de façon à éviter le problème des pôles. Ceci revient à utiliser des facettes ayant un nombre de côtés variable, rassemblées dans un graphe de contrôle, permettant ainsi de s'affranchir de grilles de contrôle régulières.

Dans l'algorithme d'évaluation (3.1.3), on subdivise la grille de contrôle en ligne, puis en

colonne, en utilisant la propriété de séparabilité des produits tensoriels de polynômes. De la relation de récurrence (4), on peut déduire, en prenant une subdivision deux fois plus fine de la grille de contrôle, dans le cas où  $p$  est égal à 2 :

$$Q_p^{n+1, m+1}\left(\frac{k}{2}, \frac{l}{2}\right) = \frac{1}{4}\left(Q_p^{nm}\left(\frac{k}{2}, \frac{l}{2}\right) + Q_p^{nm}\left(\frac{k}{2} - \frac{1}{2}, \frac{l}{2}\right) + Q_p^{nm}\left(\frac{k}{2}, \frac{l}{2} - \frac{1}{2}\right) + Q_p^{nm}\left(\frac{k}{2} - \frac{1}{2}, \frac{l}{2} - \frac{1}{2}\right)\right)$$

On peut alors utiliser un algorithme similaire à celui que l'on a vu, qui donne directement la subdivision suivante de la grille de contrôle. Ceci n'apporte en fait que peu de chose, si ce n'est la remarque que cette formule calcule le barycentre de chaque quadruplet de la grille. On construit ainsi un algorithme travaillant sur une grille non régulière (au sens du nombre de voisins de chaque noeud), à savoir un graphe de contrôle. L'étape de dédoublement (ici quadruplement) reste identique, mais ce sont les étapes de récurrence, utilisant le "caractère barycentrique" de la formule, qui permettront de gérer les carreaux dont le nombre de cotés est différent de quatre.

La figure 12 permet d'illustrer l'algorithme pour une B-spline d'ordre 3 : en trait continu non gras, est représenté le graphe initial, et en pointillé le résultat de la première récurrence (qui correspond à l'ordre 2). La première subdivision (ou seconde récurrence) est dessinée en gras.

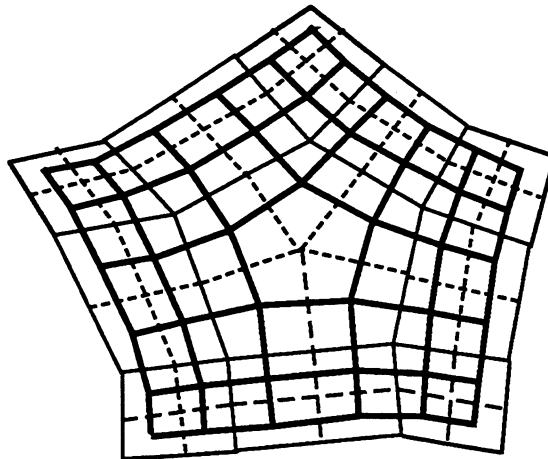


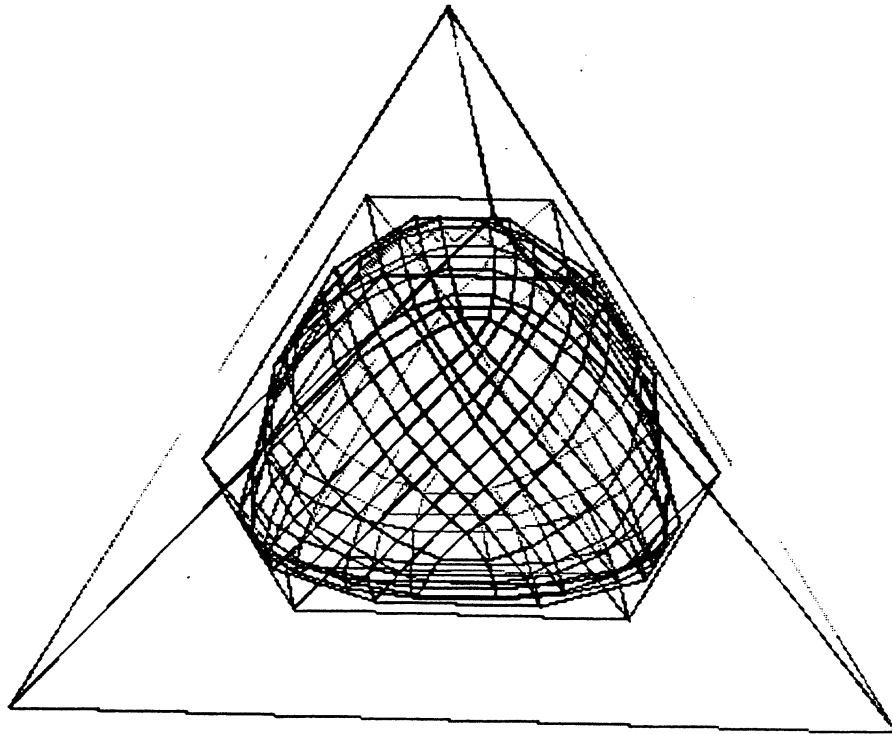
Figure 12 : schéma de subdivision d'un graphe

En fait, à chaque étape de l'algorithme, on peut trouver un ensemble de grilles régulières dans le graphe de contrôle. Les facettes à  $n$  cotés ( $n$  différent de 4) qui séparent ces grilles, convergent à l'infini en un point, et durant leur subdivision, génèrent des points de contrôle assurant le raccord continu des grilles régulières.

La figure 13 montre un exemple d'un tel algorithme, partant d'un tétraèdre de contrôle, en utilisant des B-splines d'ordre 3.

La figure 14 montre plusieurs résultats de l'algorithme de subdivision, en partant du même tétraèdre de contrôle. Dans le sens des aiguilles d'une montre, on aperçoit une surface d'ordre 2, 3, 5 et 4. Du fait que le nombre de pas de l'algorithme dépend de l'ordre des B-splines utilisées, que les surfaces d'ordre pair ne produisent que des graphes de contrôle à facettes à quatre cotés (les noeuds ont un nombre de voisins quelconques), à l'inverse des surfaces d'ordre impair.

Notons que pour des raisons de visibilité, un facteur de zoom a été appliqué pour les surfaces d'ordre élevé. En pratique, chaque surface convexe d'ordre  $p$  contient celle d'ordre  $p+1$ .



*Figure 13 : subdivisions successives d'un tétraèdre de contrôle*



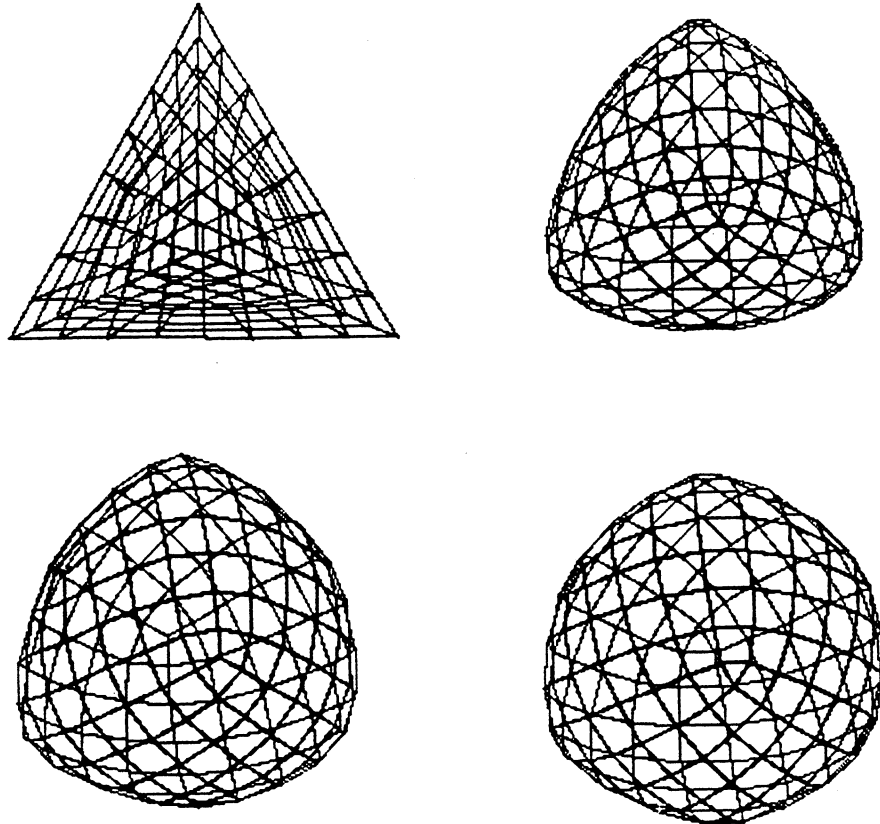


Figure 14 : surfaces d'ordre 2, 3, 4, 5 issues d'un tétraèdre

Il apparaît que cette solution de représentation est plus générale. Durant le reste du chapitre, nous étudierons différents problèmes pour les deux types de représentations, afin de pouvoir les comparer. Mettons au passif de cette dernière solution la lourdeur de la gestion des voisinages.

#### 4- Propriété de raffinement local

##### 4.1- Raffinement local d'une grille de contrôle

Sachant maintenant évaluer une courbe spline à partir de ses points de contrôle, on cherche à obtenir un raffinement de cette courbe pour qu'elle soit adaptable à volonté. Ceci revient à pouvoir disposer d'un nombre de points de contrôle en quantité suffisante à l'endroit voulu. L'intérêt de l'algorithme d'évaluation ci-dessus est qu'il fournit à chaque étape de subdivision deux fois plus de points que précédemment.

Contrairement à l'utilisation habituelle de cet algorithme, qui se limite à évaluer une spline, nous proposons de considérer que chaque étape fournit un véritable polygone de contrôle, dont chaque point peut avoir une position spatiale quelconque. Cette position est initialisée par l'algorithme de subdivision à partir du polygone initial. Cependant, rien

n'empêche de déplacer un de ces points. Bien sûr, et c'est tout l'intérêt, la courbe spline qui en sera déduite est différente de la courbe initiale, la différence portant sur les B-splines dont les supports contiennent le point déplacé.

Cette constatation montre[FOR] qu'il est possible de raffiner localement une courbe spline, même définie initialement par un polygone "grossier". Selon la taille du raffinement voulu, il faut modifier les B-splines englobant ce raffinement, ce qui impose de choisir le déplacement d'un ou de plusieurs points de contrôle aux subdivisions appropriées.

De plus, l'utilisation de cette technique est économique pour l'évolution d'une surface, puisque seuls les points ainsi raffinés auront à être déplacés, les autres étant déduits par subdivision du polygone de contrôle initial.

L'algorithme de subdivision est alors adapté au raffinement, puisqu'il suffit de conserver une liste de points "mobiles" pour chaque subdivision. Reprenons l'exemple précédent d'évaluation d'une courbe ouverte en subdivisant le polygone  $(P_0, \dots, P_3)$ . On obtient (figure 15) à partir des  $Q_2^3(i)$  le polygone de contrôle de première subdivision  $(Q_0, \dots, Q_5)$ . Mais on raffine la courbe au niveau de  $Q_4$ , en lui donnant la possibilité de se déplacer pour donner localement à la courbe un peu plus de souplesse.

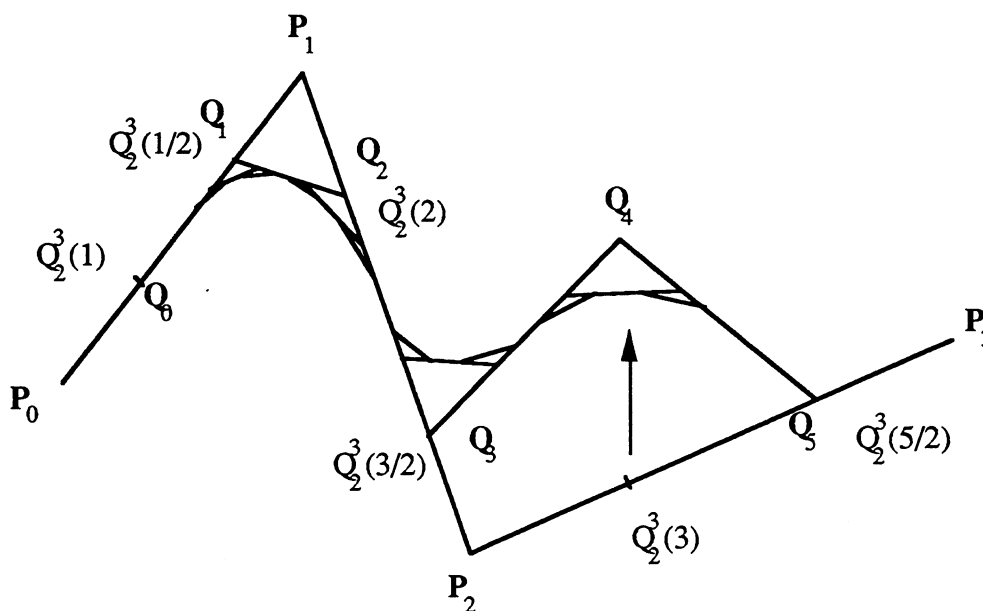
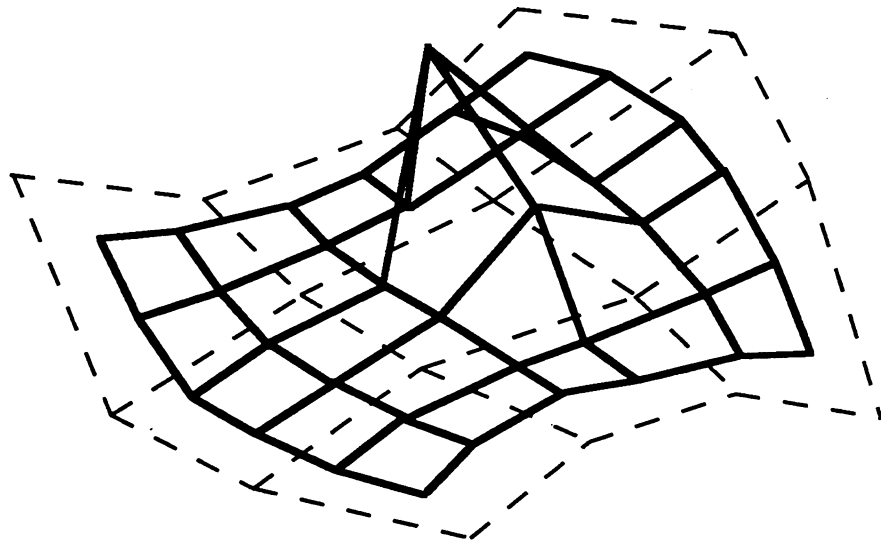


Figure 15 : raffinement au point  $Q_4$  par rapport à la première subdivision de  $(P_0, \dots, P_3)$

En effet, le raffinement a permis à la courbe, initialement définie par quatre points, de l'être maintenant par cinq ( $P_0, \dots, P_3, Q_4$ ). L'utilisation directe de tous les points de la première subdivision reviendrait à manipuler six points ( $Q_0, \dots, Q_5$ ). En fait, plus le raffinement agit sur une petite portion de la courbe (c'est à dire une forte subdivision), plus cette technique de raffinement est appropriée.

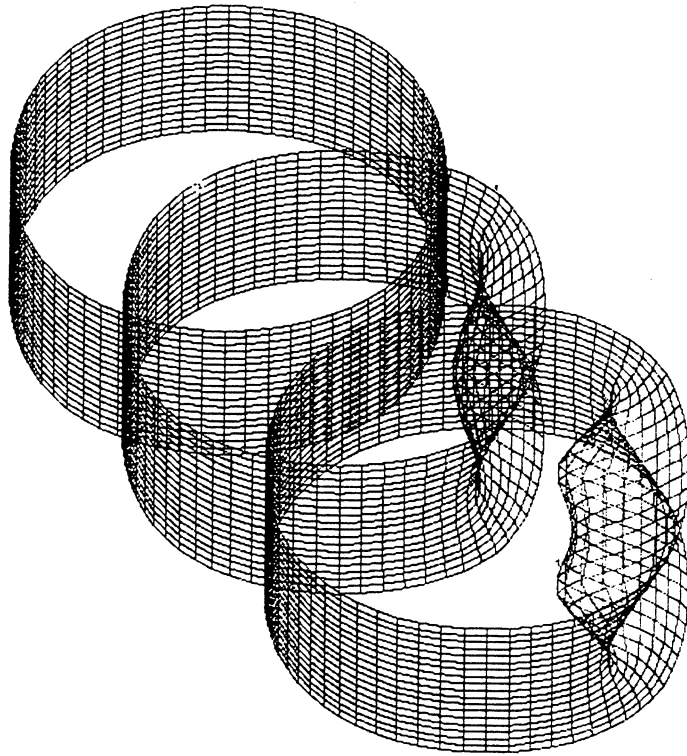
L'extension de cette technique de raffinement se fait facilement aux surfaces, puisque l'algorithme de subdivision évalue successivement des grilles de contrôle de plus en plus fines. Il suffit de connaître la taille du carreau de B-spline à raffiner pour pouvoir choisir une subdivision et rendre mobiles les points appropriés.

En reprenant la grille de l'exemple d'évaluation de surface, la figure 16 montre une possibilité de raffinement de la grille de contrôle correspondant à la première subdivision. On raffine ici la surface au voisinage de deux points : la surface est alors localement plus souple, et déduite de  $16+2$  points de contrôle. Déplacer tous les points de contrôle de la nouvelle surface reviendrait à déplacer 36 points.



*Figure 16 : raffinement d'une grille de contrôle à la première subdivision*

La figure 17 illustre le raffinement d'un cylindre en un point à la première subdivision puis en deux points à la deuxième subdivision. L'objet présente les points issus de la quatrième subdivision.



*Figure 17: deux raffinements successifs d'un cylindre*

#### **4.2- Raffinement polaire**

De la même façon que l'on a un traitement particulier quant à la construction de surfaces à pôles, le raffinement de points de contrôle sur le support d'un pôle (appelons-les points polaires) est soumis à des contraintes dues à la structure du pôle.

En effet, le raffinement d'un point de contrôle permet à ce point d'occuper une position spatiale quelconque. Or, la structure polaire retenue pour ses propriétés de continuité de la surface implique que ses subdivisions successives conservent ces propriétés. Cependant, un point raffiné comme précédemment ne les respecte plus implicitement. Aussi faut-il soit gérer de manière appropriée le déplacement des points polaires lors de l'évolution de la surface, soit trouver une contrainte pour assurer en permanence la continuité au pôle. Là encore, on souhaite décharger l'étape d'évolution de cette responsabilité, pour des raisons d'efficacité et de simplicité.

Reprenons l'exemple du pôle Nord pour faciliter les notations. Sachant que l'on a initialisé la surface fermée avec seulement trois colonnes de points de contrôle pour assurer une planarité implicite aux pôles, on souhaite retrouver cette situation. Observons

de plus que la superficie des carreaux de B-splines de la frontière Nord est d'autant plus réduite que le nombre de subdivisions est élevé, et appelons cette zone  $ZPN_s$ , où  $s$  est la subdivision courante. Raffiner un point dans  $ZPN_s$  peut être nécessaire, mais on peut très bien raffiner, à la subdivision  $s'$  ( $s' > s$ ), plus de points sur une zone équivalente, mais de façon à ce que ces points n'appartiennent pas  $ZPN_s$ . La subdivision de points polaires n'est donc pas cruciale.

Il faut cependant raffiner  $ZPN_0$  pour rendre la surface adaptable, mais on ne va raffiner que trois points seulement, les autres points étant fonction de ces trois "points directeurs". Le nombre trois est bien entendu choisi pour retrouver une coplanarité implicite. Soit  $(P_{pj})$  la suite des points polaires à raffiner,  $j \in [0, M]$  et  $p$  le numéro de la ligne de contrôle définissant le pôle. On choisit :

- les points directeurs par  $D_0 = P_{p0}$ ,  $D_1 = P_{p\frac{M}{3}}$  et  $D_2 = P_{p\frac{2M}{3}}$
- un repère polaire  $R_{p\text{pôle}} = (D_0, \overrightarrow{D_0D_1}, \overrightarrow{D_0D_2})$

Les autres points de ligne  $p$  obtiennent alors des coordonnées dans ce repère. Bien sûr, la famille  $(D_0, D_1, D_2)$  doit être libre, et il est du ressort de l'algorithme d'évolution de conserver cette condition somme toute assez simple. Notons que dans le cas contraire il y a perte de la continuité  $G^1$  mais pas  $G^0$ . Aussi confondre les trois points directeurs en un, le pôle, peut être utile le cas échéant.

La figure 18 représente la construction de ce repère avec  $p = 1$ . Les points  $P_{10}$ ,  $P_{12}$  et  $P_{14}$  forment le repère polaire  $R_{p\text{pôle}}$  et sont donc raffinés alors que les trois autres points obtiennent des coordonnées dans ce repère.

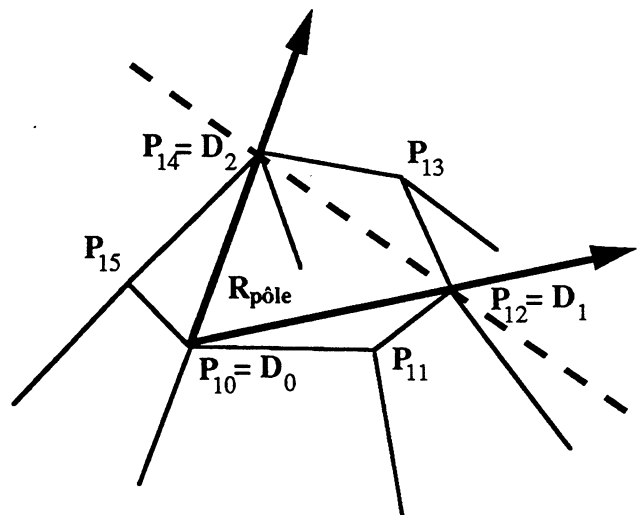


Figure 18: construction du repère polaire  $R_{p\text{pôle}}$  à partir de trois points mobiles

Compte tenu du raffinement possible des trois points directeurs, leur mouvement est donc possible, comme pour tout autre point raffiné. Cependant, les positions des autres points de la même ligne sur la grille de contrôle ne sont plus déduites par subdivision, mais à l'aide de leurs coordonnées dans  $R_{\text{pôle}}$ .

La figure 19 montre le déplacement du point directeur  $D_0$  et les transformations induites du repère  $R_{\text{pôle}}$  en  $R'_{\text{pôle}}$ . Par suite tous les autres points non directeurs de la ligne  $p$  sont déplacés.

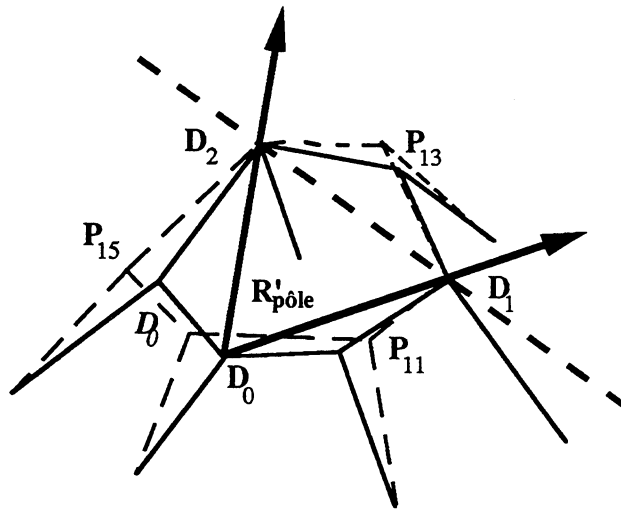


Figure 19 : déformation du repère  $R_{\text{pôle}}$  en  $R'_{\text{pôle}}$  et des points polaires

#### 4.3- Raffinement d'un graphe de contrôle

De la même façon que pour une grille régulière, puisque l'on utilise une variante de l'algorithme de subdivision, on peut raffiner un graphe de contrôle à chaque étape de la subdivision. La figure 20 montre deux raffinements sur un tétraèdre de contrôle par l'effet produit sur la troisième subdivision.

Il apparaît que la structure à base de graphe de contrôle est aussi souple d'emploi que celle utilisant une grille régulière, munie de pôle pour une surface fermée. En fait les pôles apparaissent comme une lourdeur, non négligeable par la suite.

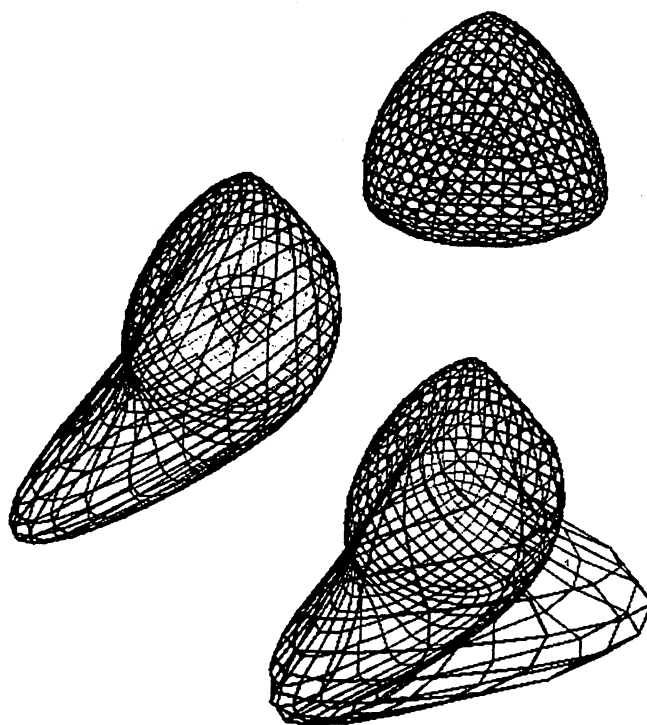


Figure 20 : deux raffinements successifs d'une structure à facettes

## 5- Auto-intersection d'une surface

### 5.1- Deux types d'intersections

Suite à l'évolution de ses points de contrôle, une surface est donc en mesure de se déformer. Aussi, la surface fermée de type cylindre à pôles va pouvoir être adaptée à une forme tridimensionnelle. Cependant toutes les formes ne sont pas de topologie équivalente à une sphère, et la surface qu'on utilise va rencontrer deux types de problèmes :

- l'objet à rejoindre est de type torique. De manière générale, il présente un ou plusieurs trous, et la surface va s'auto-intersecter en rentrant dans ces trous. Appelons "trou" ce type d'intersection.

- l'objet n'est pas connexe et est donc composé de plusieurs sous-objets, eux-mêmes de topologie inconnue. La surface va aussi s'auto-intersecter en "s'infiltrant" entre ces objets et appelons "étranglement" ce type d'intersection.

Anticipons sur la méthode de déplacement des points de contrôle et précisons que l'évolution de la surface se fera en modélisant une tension surfacique orientée selon ses propriétés de convexité locale. Dans un cas, la partie de la surface entrant dans un trou correspond à une zone concave, et produira une intersection ponctuelle avec son

symétrique entrant par l'autre côté du trou. Dans l'autre cas, la partie de la surface s'infiltrant entre deux objets présente une suite de points de type "col" (au sens de la courbure, ce qui sera précisé plus loin), qui cherchera à se confondre en un point.

Dans les deux cas de figure, l'intersection reste ponctuelle, puisque dans le premier cas deux surfaces concaves vont d'abord s'intersecter en un point (on verra dans le chapitre 4 comment ramener le cas général à une intersection ponctuelle), et dans le deuxième cas, les points cols vont s'agglutiner par construction en un point.

Aussi, la résolution d'une auto-intersection de surface doit se ramener à un changement de topologie de cette surface. Ce changement est réalisable en modifiant la structure de contrôle initiale. En effet, on ne peut définir plusieurs objets, ou un objet avec plusieurs trous, à l'aide d'un seul produit tensoriel de B-splines. Il s'agit alors de déduire plusieurs grilles de contrôle appropriées à partir de la première. Le principe consiste à réordonner les voisinages des points d'intersection de manière à raccorder ou à séparer des éléments de surface. Ces raccords devenant explicites, il faut alors briser la structure de produit tensoriel global en autant de produits tensoriels que nécessaire.

## **5.2- Le type "trou"**

### **5.2.1- Cas d'une grille régulière**

De manière générale, le type d'intersection étudié ici a pour but de créer un trou quand deux surfaces entrent en contact de manière ponctuelle.

Sachant qu'une surface est approximée par une grille de contrôle, on utilise l'ordre sur cette grille pour associer aux deux surfaces un repère, appelé repère surfacique. Les coordonnées d'un point dans ce repère seront donc les coordonnées du point de la grille qui l'approxime au mieux (notons que, compte tenu de l'évaluation discrète des surfaces, il y a identité entre les points de la grille et les points de la surface à partir d'une certaine subdivision).

Connaissant la position du point intersection sur la surface, on peut trouver une subdivision pour laquelle un point de contrôle approxime convenablement ce point. La création du trou se fait de manière symétrique pour les deux surfaces : si  $i_0$  et  $j_0$  sont les coordonnées paramétriques du point intersection, on peut couper les suites de noeuds  $(u_i)$  et  $(v_j)$  en quatre sous-suites, définissant ainsi quatre grilles continûment raccordées.

Pour raccorder les deux surfaces, on va donc raccorder les quatre sous-suites avec leurs homologues au niveau des carreaux de surface définissant le point d'intersection. D'un



point de vue pratique, les raccords explicites à réaliser demandent bien sûr le raffinement de tous les points impliqués, ce qui n'est pas gênant : le trou étant ponctuel, il faudra l'adapter par la suite à l'objet.

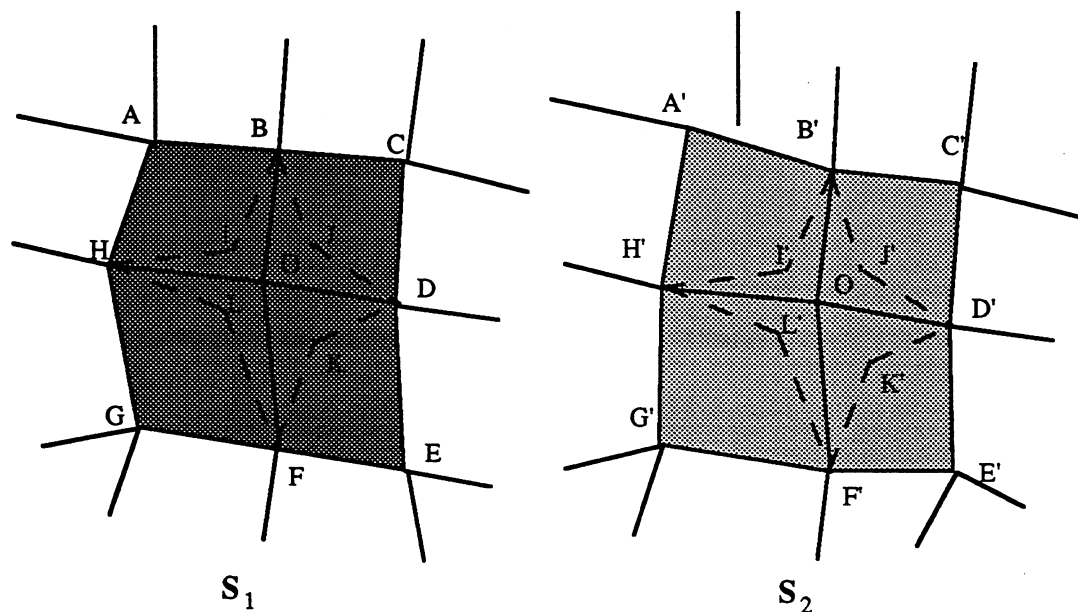


Figure 21 : carreaux en contact en O avant construction du trou

Pour expliquer le principe de la création du trou, on utilise dans les figures suivantes un exemple de surfaces d'ordre 2 : il y a alors identité entre les grilles de contrôle, et la représentation des surfaces elles-mêmes. La figure 21 montre les deux surfaces, et quatre carreaux définissant le point d'intersection O. Soient I, J, K, L les points correspondant à O sur chacune des sous-grilles.

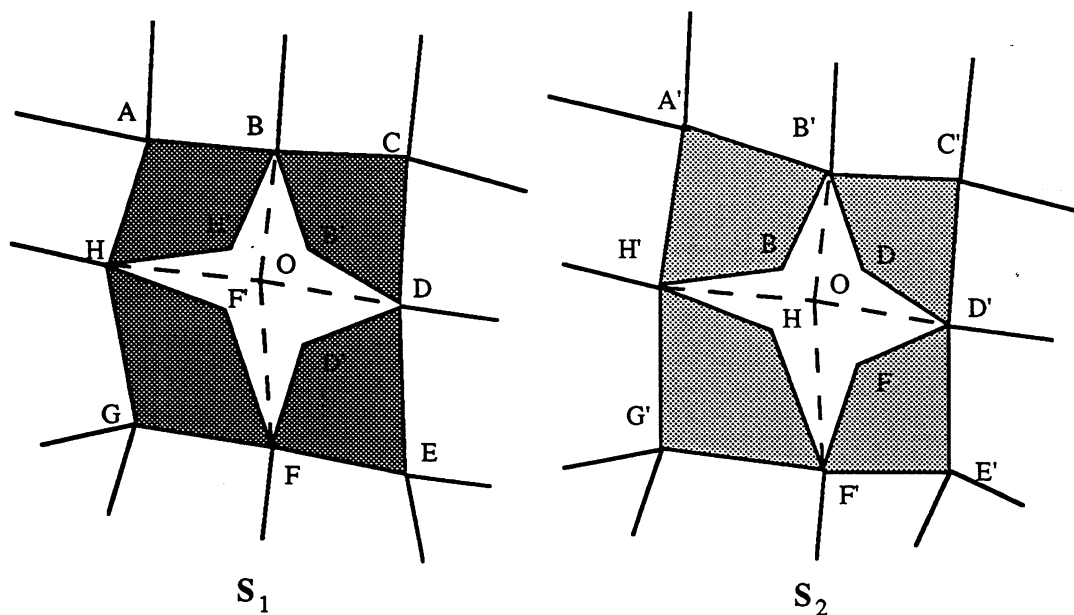


Figure 22 : réordonnancement des raccords pour la construction d'un trou

Il est alors possible (figure 22) "d'ouvrir" en O la surface  $S_1$ , le long des bords des sous-grilles. Les bords ainsi ouverts sont ensuite raccordés à nouveau, mais avec leurs homologues (figure 22), moyennant une "rotation de  $45^\circ$ ". Chaque "coin" des sous-grilles de  $S_1$  (par exemple B), va se placer à la jonction de deux sous-grilles de  $S_2$  (B').

La figure 23 constitue une représentation du réordonnancement, les carreaux gris foncé représentant les coins des quatre carreaux de  $S_1$ , et les carreaux gris clair ceux de  $S_2$ , disposés dans l'ouverture faite dans  $S_1$ . Les liaisons des carreaux avec leurs surfaces respectives sont dessinées en trait plein pour  $S_1$ , et en pointillés pour  $S_2$ .

Pour des surfaces d'ordre supérieur à 2, le principe est identique au niveau du positionnement des sous-surfaces. Les sous-grilles de contrôle se recouvrent au départ continûment, et déterminer un recouvrement équivalent après le réarrangement, oblige certains points de contrôle à être déterminés en fonction de leurs voisins. Aussi, on n'impose qu'une continuité  $G^0$  aux sous-surfaces en présence. Pour cela, on dégénère en un point les patches contenant B et B' (et similaires). Ceci entraîne donc la création de huit points où la continuité est  $G^0$ . C'est en fait sans importance, puisque l'on pourra ensuite raffiner des points de subdivision supérieure.

La figure 24 montre un exemple d'un tel raccord entre deux surfaces d'ordre 3.

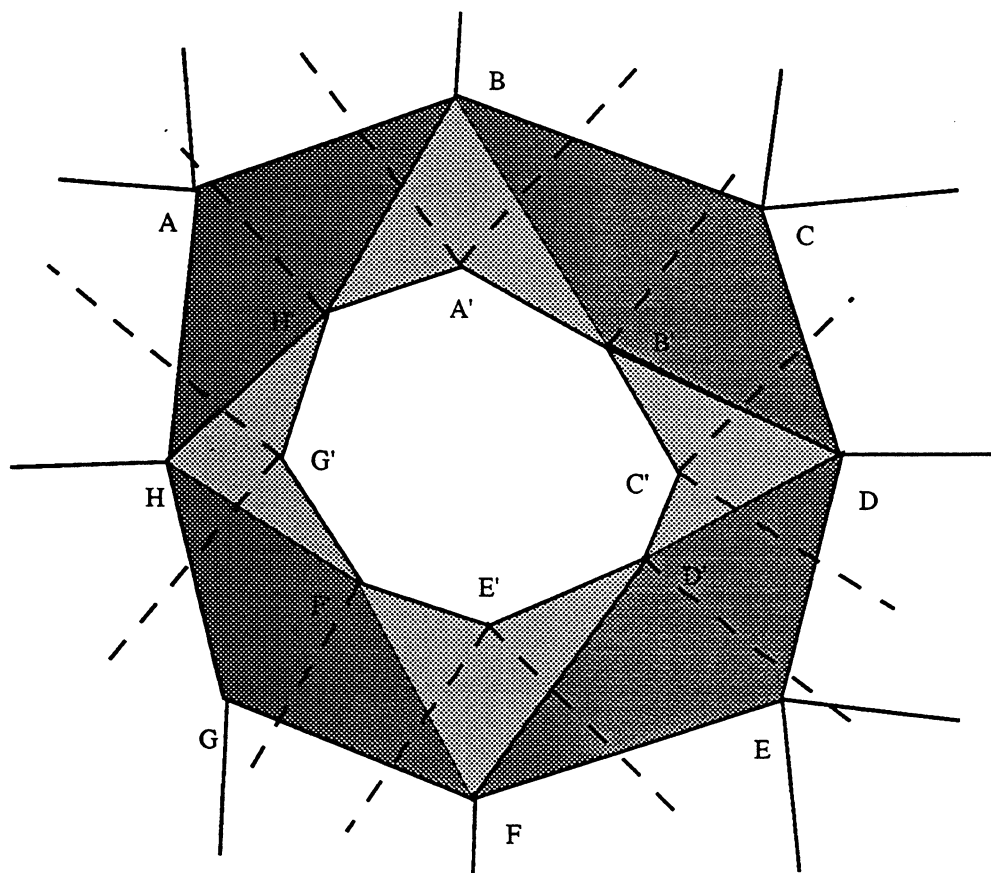
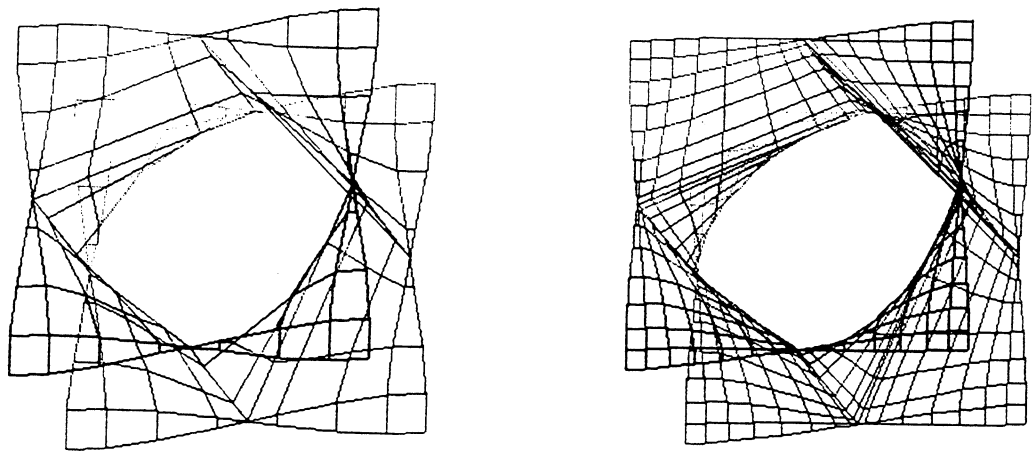


Figure 23 : représentation du raccordement en trou

Une limitation de cette méthode consiste en la nécessité d'avoir un nombre de noeuds dans chaque sous-suite compatible avec l'ordre  $p$  de la B-spline utilisée. En effet, si  $n$  est le nombre de noeuds en colonne de la grille initiale, et si le point  $O(i_0, j_0)$  est tel que  $n - i_0 < p$ , une au moins des sous-suites générées par le trou n'a pas assez de noeuds (sauf si la grille est périodique, par exemple un cylindre) pour définir une spline d'ordre  $p$ . Dans ce cas, il suffit d'utiliser une subdivision supérieure de la grille et d'effectuer le raccord sur le point homologue à  $O$ .



*Figure 24 : raccord entre deux surfaces à deux subdivisions différentes*

On a traité le cas le plus général, mais que se passe-t-il dans le cas des pôles?

Le premier cas revient à mettre en contact deux pôles. La solution est alors évidente : il suffit de supprimer les deux pôles et de recouvrir mutuellement les deux grilles ainsi "ouvertes".

Le deuxième cas, intermédiaire, met en contact un pôle avec un point quelconque d'une grille. On groupe les carreaux du bord Nord du produit tensoriel, en quatre carreaux dégénérés en quatre points. Ces carreaux vont alors jouer le même rôle que leurs homologues de l'autre surface.

### 5.2.2- Cas d'un graphe de contrôle

La création d'un trou dans un graphe de contrôle ne présente aucun problème particulier, puisqu'il est constitué de noeuds ayant un nombre de voisins quelconques. Soient  $P$  et  $P'$

les points du trou, et soient  $\text{card}_P$  et  $\text{card}_{P'}$  ( $\text{card}_P \leq \text{card}_{P'}$ ) le nombre de facettes contenant respectivement  $P$  et  $P'$  dans le graphe. On "ouvre" la surface en dupliquant  $P$ ,  $\text{card}_P$  fois. Ces points contrôlent alors le trou, et forment une *couronne* (bien qu'ils soient spatialement confondus). Dans les facettes contenant  $P$ , on remplace l'occurrence de  $P$  par deux points successifs de la couronne (on définit un ordre sur la couronne). Pour les facettes contenant  $P'$ , si  $\text{card}_P$  est inférieur strictement à  $\text{card}_{P'}$ , l'occurrence de  $P'$  peut n'être remplacée que par un seul des points dupliqués, de façon homogène avec le rapport  $\text{card}_P/\text{card}_{P'}$ .

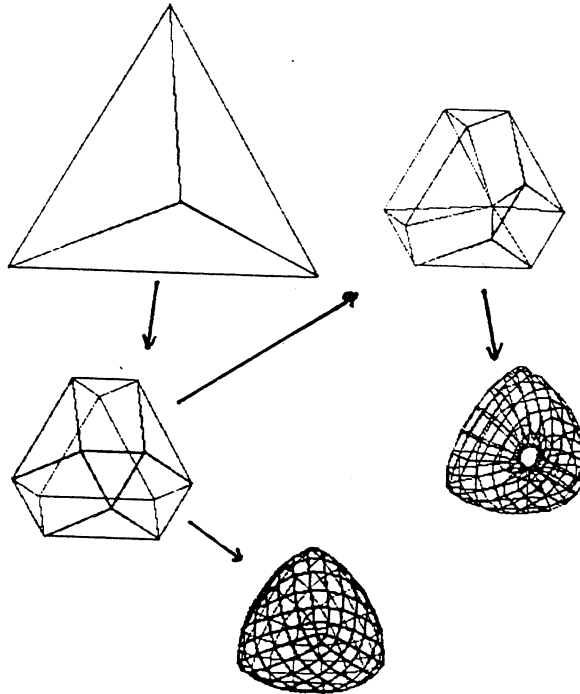


Figure 25 : création d'un trou dans une structure à facettes

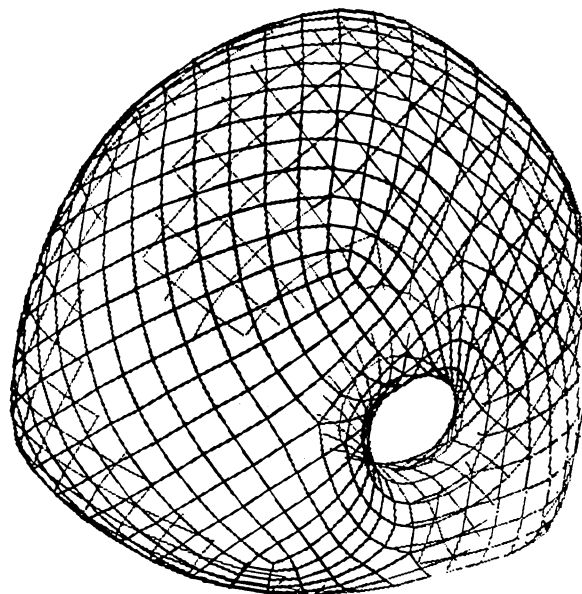


Figure 26 : exemple d'une surface à trou d'ordre 4

Dans la figure 25, on montre un tétraèdre de contrôle initial. La première subdivision est située en dessous, et sur celle-ci, on choisit de "percer" l'objet en un point, d'où le deuxième graphe de contrôle en subdivision 1. On voit ensuite les résultats avec deux subdivisions de plus des deux graphes.

Cette méthode est bien sûr indépendante de l'ordre des B-splines utilisées, puisque c'est une simple modification du graphe de contrôle. La figure 26 présente un trou réalisé dans une surface d'ordre 4, et on peut noter, en plus des points à trois voisins issus du tétraèdre, des points à cinq voisins issus du passage en topologie torique.

Notons que le changement de topologie sur les graphes de contrôle, est de loin plus aisé que sur les grilles régulières. Il conserve de plus une bonne continuité, qui est plus difficile à obtenir dans le cas régulier.

### **5.3- Le type "étranglement"**

La définition d'un étranglement de surface est assez bien représentée par l'image d'un sac que l'on ferme avec une corde. En fait, on verra dans le chapitre 4 que les contraintes de contraction qu'on impose à une surface déformable, contenant deux objets disconnexes, provoqueront cet aspect de "sac".

On caractérise un étranglement par le resserrement d'une surface fermée en un point et un seul, ce point séparant spatialement la surface en deux surfaces  $S_1$  et  $S_2$  fermées et topologiquement disjointes. Une généralisation de cette définition pourrait être que ce point sépare spatialement la surface en un nombre dénombrable de surfaces fermées. Dans la suite on suppose que le nombre de parties obtenues à partir de la surface initiale, vaut deux, dans un but explicatif.

#### **5.3.1- Cas d'une grille régulière**

Un repère étant associé à la surface, séparer spatialement deux parties de cette surface implique que les points frontière entre ces parties sont en fait confondus sur le point d'étranglement. Ceci assure que l'ensemble des points frontière est connexe et de dimension un dans l'espace paramétrique. Cette propriété pourra être utilisée pour la détection des points de l'étranglement.

Supposons cet ensemble connu, et voyons maintenant comment séparer physiquement ces deux surfaces, au niveau des points frontière. Les figures 21 et 22 représentent dans

un cas simple la réalisation d'un étranglement : les traits gras représentent les limites des carreaux de  $S_1$  qui n'appartiennent pas à la frontière de l'étranglement (laquelle est dessinée en pointillés).

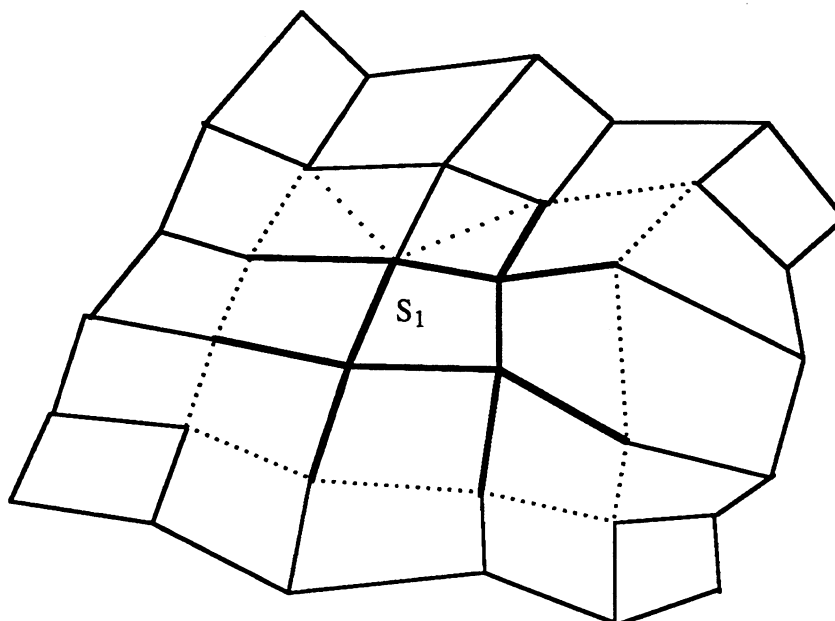


Figure 27 : grille de contrôle initiale

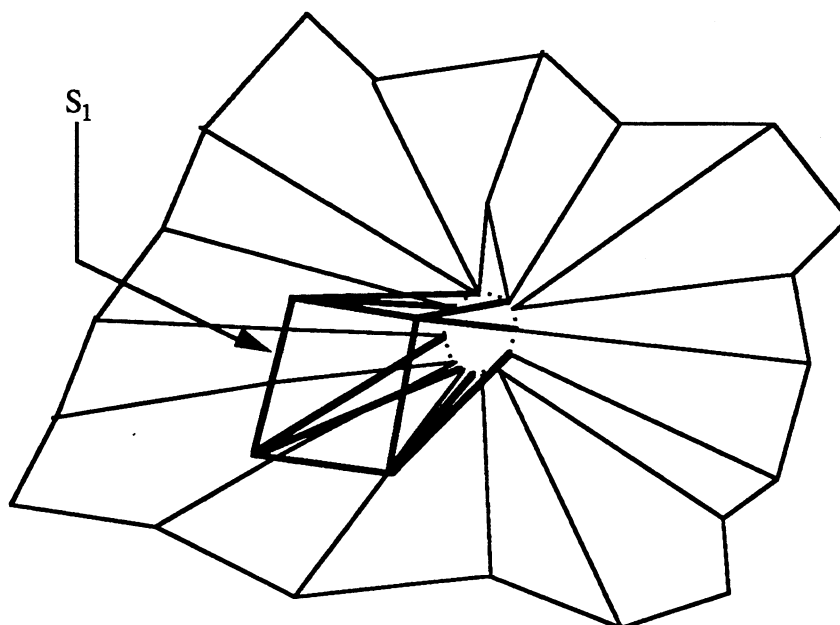


Figure 28 : grille de contrôle à l'étranglement

Dans les schémas présentés ici, il apparaît que le lieu des points, sur la grille de contrôle, des deux zones séparées par l'étranglement, n'est pas rectangulaire. Dans le cas général, il n'y a d'ailleurs aucune raison pour qu'il le soit. Aussi, alors que dans le cas du trou, on conservait une structure de produit tensoriel, il va falloir ici trouver une méthode pour

évaluer séparément les objets issus d'un étranglement.

Pour cela, on utilise la solution, peu élégante, qui consiste à dupliquer la structure initiale en autant d'objets qu'il y a de parties créées. Puis, dans chacun de ces nouveaux objets, on va "masquer" les points qui n'appartiennent pas à la partie considérée. Ceci est réalisé en raffinant systématiquement tous ces points en un seul point, qui a pour coordonnées le "point de coupure" de l'objet initial.

On veut maintenant étudier la continuité des surfaces obtenues. Il apparaît qu'elles conservent les mêmes propriétés que la surface initiale, sauf au point de coupure, où les surfaces sont  $G^0$ . Il reste possible d'obtenir une continuité  $G^1$  en ce point, en utilisant une méthode similaire à la construction des pôles pour un cylindre. La coupure entre les deux parties peut très bien être associée au bord d'un cylindre à refermer, et on peut avoir le même raisonnement que dans le cas de la construction du pôle. On impose cependant seulement une continuité  $G^0$  en ce point, ce qui ne gênera pas la suite du processus, si on peut raffiner des points aux subdivisions suivantes.

La figure 29 montre un exemple d'une telle séparation. Partant d'un cylindre à pôles (vue du haut), on effectue un étranglement sur les points de l'équateur à la subdivision 1. La vue du bas montre la surface associée à cette grille de contrôle. Il est cependant possible de "couper" cet étranglement, et d'obtenir deux objets différents (vue de droite). Remarquons la "pointe"  $G^0$  après coupure.

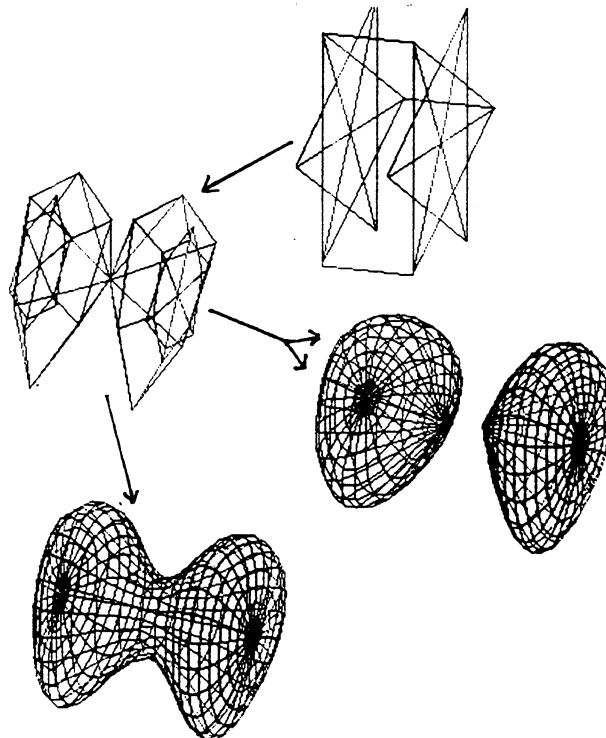


Figure 29 : cylindre à pôles sectionné à l'équateur à partir de la première subdivision

Que se passe-t-il si un pôle se trouve au point de coupure? La structure polaire consiste essentiellement à garder la surface fermée, et à conserver une bonne continuité au pôle. Si le pôle doit être coupé par un étranglement, ces propriétés n'ont plus lieu d'être, et on supprime simplement cette structure. En effet les points du pôle seront tous dégénérés sur le point de coupure.

Un dernier cas particulier apparaît : comment se comporte un tel objet si le point de coupure entre en contact avec un autre point, pour faire un trou? Comme la grille initiale avant étranglement a été conservée, il reste toujours possible de déterminer quatre sous-grilles. La seule différence vient du fait que les points situés au niveau du trou sont raffinés en un seul point. Aussi, comme on l'a vu dans le paragraphe précédent, va-t-on modifier ces raffinements de telle sorte que les coins des quatres sous-grilles soient raffinés en quatre points différents, de même position spatiale.

### 5.3.2- Cas d'un graphe de contrôle

De la même manière que pour le trou, on va voir que la section d'un objet sur un étranglement se résoud facilement. Connaissant les zones du graphe séparées par cet étranglement, il suffit de séparer les sous-graphes, les points frontières étant dupliqués sur chacun de ces sous-graphes. Le seul traitement particulier consiste à fermer les bords des sous-graphes : en fait, une facette définie par tous les points frontière convient.

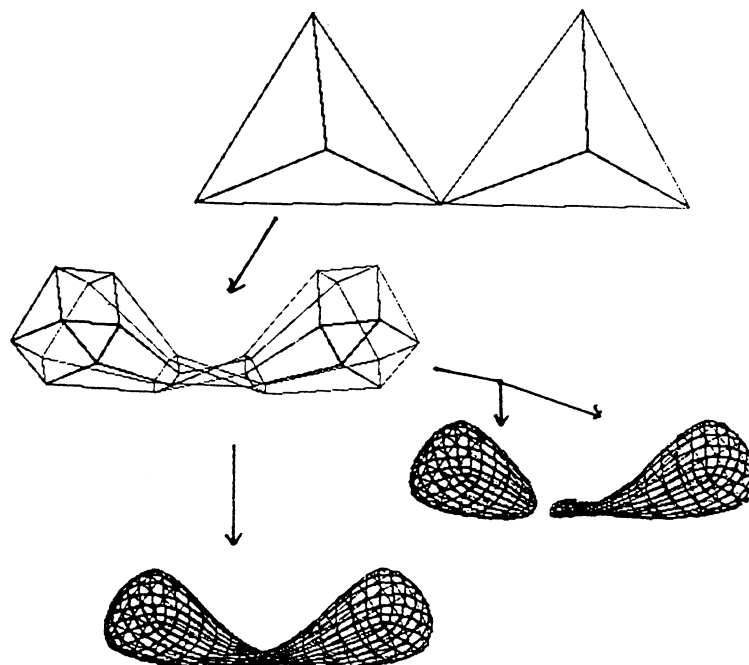


Figure 30 : double tétraèdre de contrôle sectionné à la première subdivision



Dans la figure 30, on réalise un étranglement en reliant deux tétraèdres (technique du trou). A la première subdivision de ce tétraèdre siamois, on décide de couper l'étranglement. Enfin suivent une vue de la surface sans coupure, et une vue séparant les deux objets.

A l'issue de ce paragraphe, on s'aperçoit que la représentation sous forme de graphe de contrôle est de loin supérieure à celle sous forme de grilles régulières, en ce qui concerne les changements de topologie, notamment dans le cas de la coupure d'un objet en plusieurs. La continuité de la surface initiale est préservée, et le réarrangement des points plus aisé à réaliser.

## **6- Mise en œuvre logicielle**

### **6.1- Codage des coordonnées des points**

On a vu que l'algorithme de subdivision se base sur le choix d'une suite de noeuds  $p$  fois plus fine. Dans les calculs des coefficients apparaissent alors des divisions par  $p$ . Aussi on prendra par la suite  $p$  égal à 2 pour des raisons de simplicité de calcul : en travaillant sur des entiers, la multiplication et la division par 2 ne représentent que des décalages.

D'autre part, on verra par la suite que les courbes sont évaluées dans un espace discret de 512 pixels d'arête. Aussi, représenter directement les points de contrôle dans des entiers présente, suite aux nombreuses divisions par deux, une grande imprécision. Pour remédier à cela et sachant qu'un codage direct n'utilise que 9 bits sur les 32 d'un entier, il suffira d'effectuer un décalage initial de 20 bits sur la gauche. On obtient alors une précision considérable.

### **6.2- Evaluation des surfaces**

Dans ce paragraphe seront exposées les caractéristiques de réalisation logicielle. En particulier, on détaillera la structure de données utilisée pour la représentation des surfaces.

#### **6.2.1-Evaluation d'une grille de contrôle**

On a vu que l'algorithme de subdivision d'une surface de contrôle se faisait en deux passes successives en ligne et en colonne. Pour chacune de celles-ci, il y a duplication

des points de contrôle, puis utilisation de la relation de récurrence pour la subdivision, à concurrence de la valeur du degré de la B-spline.

Dans le cas d'une courbe, les noeuds sont rangés dans le tableau  $P_1$ , et la première étape les duplique. L'algorithme faisant évoluer les noeuds deux à deux, il est alors nécessaire d'introduire un deuxième tableau  $P_2$ . Il apparaît d'ailleurs (figure 31) que, selon la parité du degré, le résultat se trouve dans  $P_1$  (cas pair) ou dans  $P_2$  (cas impair). A la subdivision suivante, on repart du tableau contenant le résultat. Aussi l'évaluation finale dépend de la parité, du degré et du nombre de subdivisions.

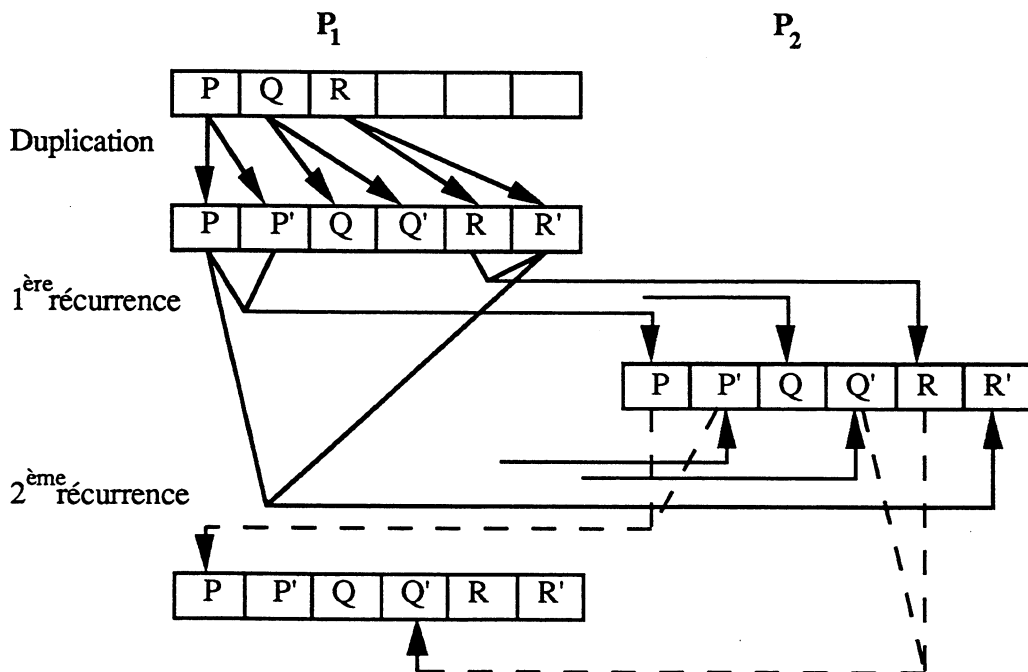


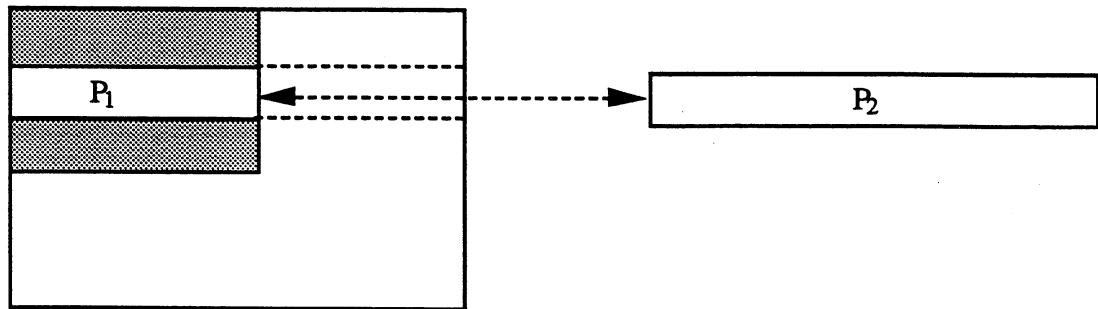
Figure 31 : mise en œuvre de l'algorithme de subdivision

Pour être sûr d'obtenir le résultat dans  $P_1$ , il suffit de dupliquer les points de  $P_1$  dans  $P_2$  plutôt que de  $P_1$  dans lui-même, dans les cas correspondants. L'évaluation d'une courbe ouverte repose sur le même principe.

Dans le cas de l'évaluation des surfaces, on utilise la procédure de subdivision ci-dessus comme brique de base, et on l'applique sur une structure bidimensionnelle. Partant de la grille initiale  $G$ , on associe  $P_1$  à toutes ses lignes, puis à toutes ses colonnes (figure 32).

G

étape ligne



étape colonne

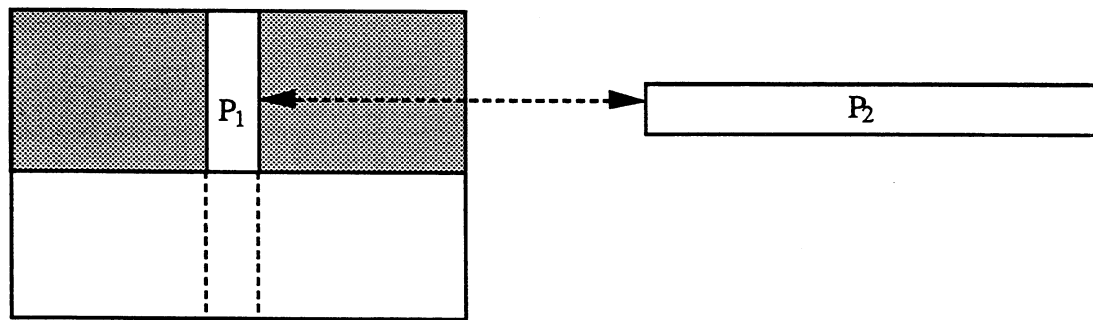


Figure 32 : passes de subdivision ligne et colonne pour une surface

### 6.2.2- Evaluation d'un graphe de contrôle

Un graphe de contrôle est défini par:

- une liste de points, ayant chacun une liste de facettes le contenant.
- une liste de facettes, définies par une liste de points.

L'étape d'évaluation consiste à appliquer sur la liste de points, la formule de récurrence établie au paragraphe 3.3.

On néglige l'étape de duplication des points, on déterminant directement le graphe subdivisé pour l'ordre 2. On obtient ainsi en plus, un point au milieu de chaque arc du graphe, et un au barycentre de chaque facette. De nouvelles facettes sont alors créées en fonction de ces nouveaux points.

Pour obtenir un graphe subdivisé correspondant à l'ordre p des B-splines utilisées, on exécute p-2 fois l'étape suivante : un nouveau graphe est créé, chaque nouveau point se

situant au barycentre des anciennes facettes. Les nouvelles facettes relient alors ces points.

Pour les ordres supérieurs à deux on utilise de plus une liste de points et de facettes auxiliaires pour effectuer les étapes ci-dessus.

Il faut souligner un élément qui n'apparaît pas dans l'algorithme : la surface étant amenée à être déformée, on va devoir calculer la normale et la convexité locale en chacun des points. Aussi les points référenceront leurs facettes voisines selon un certain ordre, et de même pour les facettes. Par suite, les étapes de subdivisions devront être écrites de telle sorte qu'elles conservent cet ordre.

### **6.3- La structure surface**

#### **6.3.1- Représentation par grilles régulières**

On est donc en mesure d'évaluer n'importe quelle grille de contrôle, et la réalisation des contraintes de raccordement, la gestion du pôle sont à effectuer avant cette évaluation. Pour cela, et surtout pour gérer les raffinements de points à chaque subdivision, on construit une structure de données adaptée (figure 33), contenant les informations relatives aux raccordements implicites, aux pôles et aux zones de raffinement. De plus, on fait une distinction entre un point évalué et un point de contrôle mobile.

La structure surface va donc contenir les éléments suivants :

- des paramètres utilisés pour l'évaluation, à savoir le degré de la B-spline utilisée, le type de raccordements implicites sur le produit tensoriel (plan, cylindre Nord-Sud, cylindre Est-Ouest, tore), et le nombre de subdivisions maximum à faire sur cette surface.

- une grille de contrôle correspondant au produit tensoriel initial. Cette grille ne contient que des adresses de coordonnées et non les coordonnées elles-mêmes des points. Ce champ n'est en fait utilisé que pour les structures surfaces "racine", les autres héritant de leur parent la position de leurs points de contrôle. Les variations sont alors introduites par des raffinements.

- une table de raffinements avec autant d'entrées qu'il y a de subdivisions possibles. A chaque entrée est associée une liste d'éléments définissant les raffinements. Un élément contient donc les coordonnées dans la grille à la subdivision courante du point raffiné, et l'adresse des coordonnées de ce point.

- une grille finale permettant de recevoir les points de contrôle et d'effectuer le calcul de subdivision (c'est la grille G précédente). Chaque cellule de cette grille contient donc un triplet de coordonnées. C'est cette grille qui contiendra les points de la surface approximée à une certaine subdivision.

Surface :

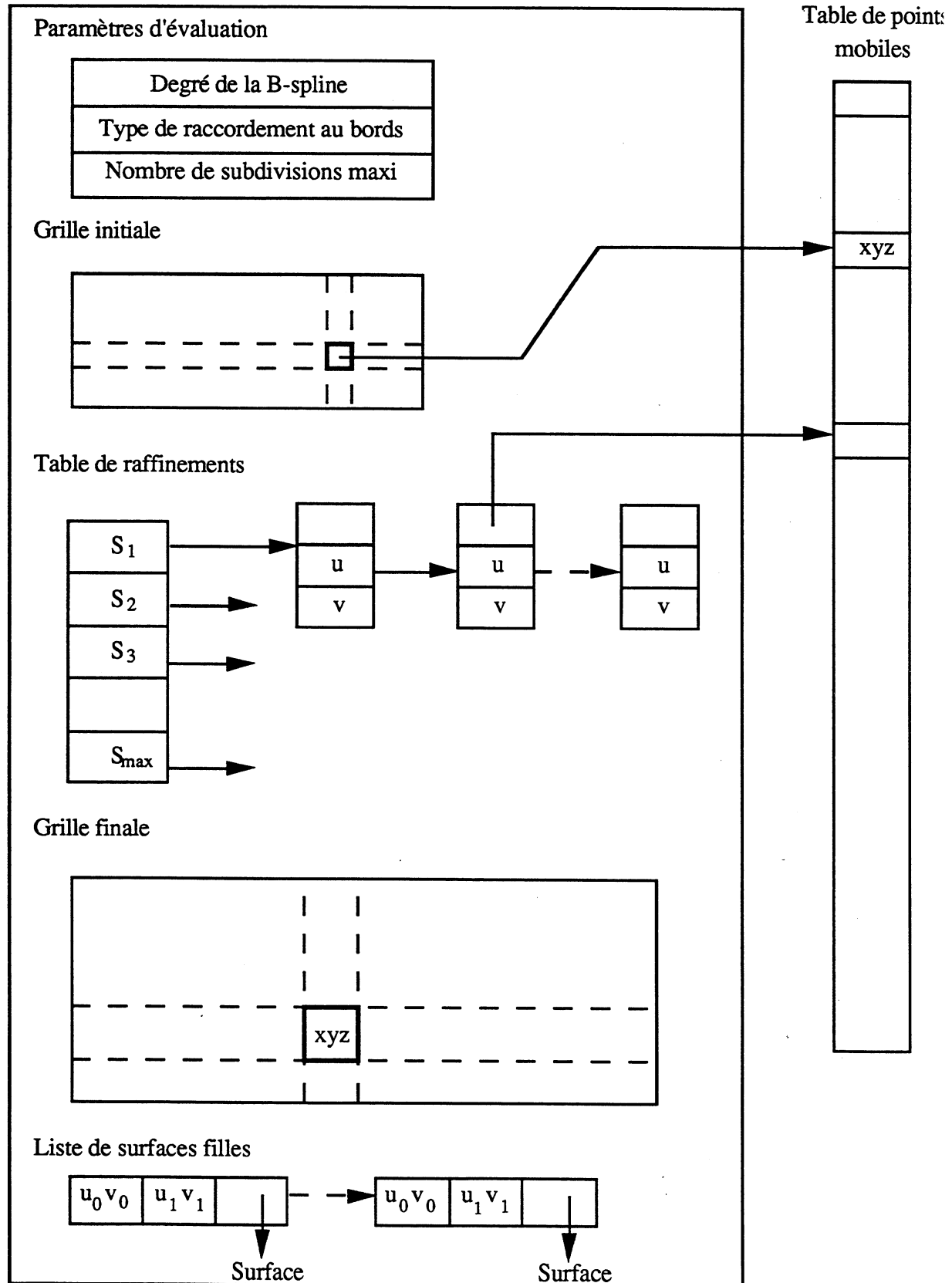


Figure 33 : structure de données d'une surface déformable

- une liste de surfaces utilisée en cas de trou ou de "sur-subdivision" dans la surface. Chaque élément contient les coordonnées des coins supérieur gauche et inférieur droit des sous-grilles, ainsi que l'adresse de la structure surface correspondante.

On ajoute à cette structure une table de points mobiles, contenant les coordonnées de ces points. Elle est référencée par les points de contrôle initiaux et tous les points raffinés. L'intérêt de cette indirection réside dans le fait que deux surfaces peuvent utiliser des points de contrôle communs dans leurs raccordements, et le déplacement d'une surface entraîne le déplacement de ses points de contrôle : la modification des points "raccord" sera alors faite de manière transparente sur l'autre surface, et permettra de conserver les propriétés du raccord sans traitement particulier.

De la même façon, la représentation du pôle entraîne des points confondus, et le déplacement de tels points est assuré par le biais de cette table : tous les points réels confondus en un point référencent en fait la même adresse, et restent toujours confondus quelles que soient les déformations de la surface.

L'évaluation d'un objet au moyen d'une telle structure revient donc à évaluer la grille de contrôle présente dans la surface "racine", et à y inclure les raffinements au fur et à mesure que l'on subdivise. On subdivise jusqu'au nombre de subdivisions maximal, à partir duquel on peut éventuellement avoir à gérer un trou. Dans ce cas, on va recopier certains éléments de la grille finale dans les grilles finales de chaque surface fille. Puis on évalue de la même manière chacune de ces surfaces, en raffinant d'abord la grille ainsi recopiée. Ce premier raffinement correspond justement aux points de contrôle du trou.

La notion de "sur-subdivision" intervient quand la subdivision d'évaluation initialement choisie n'est pas suffisante pour approximer correctement l'objet. Ceci intervient quand un raffinement apparaît à une subdivision proche de la subdivision d'évaluation. Un fort déplacement par rapport aux coordonnées qu'il aurait eues s'il n'était pas raffiné différencie sensiblement le polyèdre de la surface qu'il approxime. Inversement, pour représenter une zone localement très "torturée" de l'objet, le nombre de points de contrôle nécessaires peut excéder les points de contrôle localement disponibles.

On "sursubdivise" donc localement une sous grille de contrôle dans la zone correspondante, en rajoutant une sous-surface à cet endroit.

La figure 34 montre une grille régulière d'ordre 3, supposée être la subdivision  $s$  d'une grille initiale, sur laquelle on définit deux zones à sur-subdiviser. Dans l'angle, on subdivise plus finement un carreau, et au milieu, on remplace deux carreaux à la subdivision  $s$  par une nappe plus fine. Le raccordement entre les sous-surfaces et la grille

de subdivision  $s$ , est réalisé en dupliquant les points du bord de la sous-surface, à concurrence de l'ordre moins un de la B-spline utilisée.

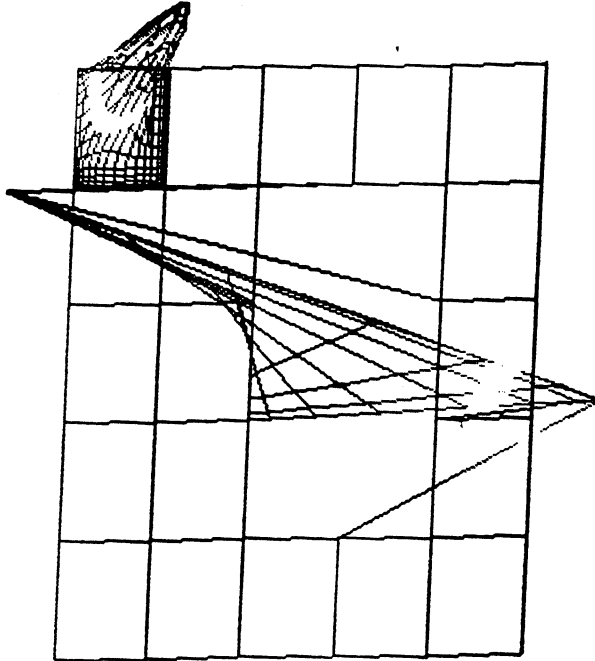


Figure 34 : élément de surface sur-subdivisé en deux zones

### 6.3.2- Représentation par graphe

Cette représentation requiert de gérer les voisinages explicites de tous les points du graphe, et on utilise la représentation indiquée à la figure 35. La référence à l'objet permet de déterminer l'appartenance du point considéré. On prévoit naturellement la position spatiale du point, et le nombre de facettes auxquelles il appartient. Enfin, on stocke une liste de référence à ces facettes. Pour cela, on alloue systématiquement quatre références contigües, puis une liste de références. En effet, on a vu que, à l'infini, les points ayant quatre voisins sont les plus nombreux. Ceci permet d'allouer d'un coup l'espace mémoire associé aux points de contrôle, en rajoutant de temps en temps une référence à une facette.

La figure 36 montre comment sont allouées les facettes. C'est donc sensiblement la même structure, mais référençant des points. De plus, on dispose d'une référence sur un point, étant le barycentre de la facette. Ce champ est utilisé dans l'étape de subdivision, pour permettre la construction des nouvelles listes de facettes et de points. Au niveau de

l'allocation mémoire des références de points, on utilise la même méthode, avec 4 champs plus une liste.

Point :

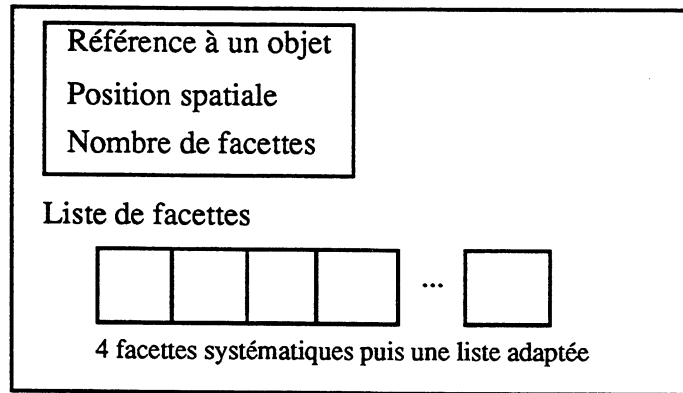


Figure 35 : structure de données "Point"

Facette :

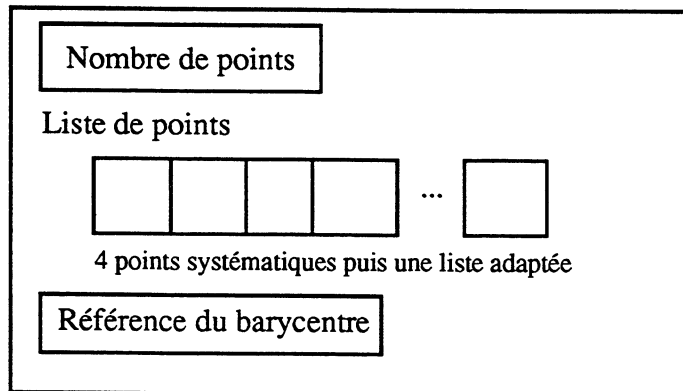


Figure 36 : structure de données "Facette"

Le graphe de contrôle contient une liste de points et une liste de facettes tels que l'on vient de les détailler. On ajoute bien sûr une liste de raffinements, destinés à adapter la surface dans son évolution future. La figure 37 permet de comparer la différence de structure entre le modèle à base de grilles régulières et celui du graphe de contrôle.

La structure de graphe ne contient donc pas de liste de points mobiles, puisque les surfaces à base de graphe n'auront pas à être raccordées explicitement. En effet, il n'existe pas de sous-surfaces à gérer, puisqu'elles sont implicitement raccordées au moyen de facettes à  $n$  cotés ( $n$  différent de quatre).

Pour terminer, voyons comment sur-subdiviser un graphe de contrôle, comme on l'a vu pour les grilles. Pour celles-ci, il est obligatoire d'adopter cette technique, en utilisant une



sous-surface plus raffinée. Ici, on va simplement utiliser l'irrégularité du graphe en se contentant d'ajouter un point dans celui-ci. De même que pour le reste des transformations du graphe, il faudra conserver pour chaque point l'ordre des facettes (et vice versa pour les facettes).

La figure 38 montre un exemple d'un tel ajout de point. A partir de la subdivision 1 du tétraèdre habituel, on choisit de rajouter deux points de contrôle. On voit ensuite l'effet sur la surface de droite, celle de gauche étant la suite de la subdivision "classique".

Surface :

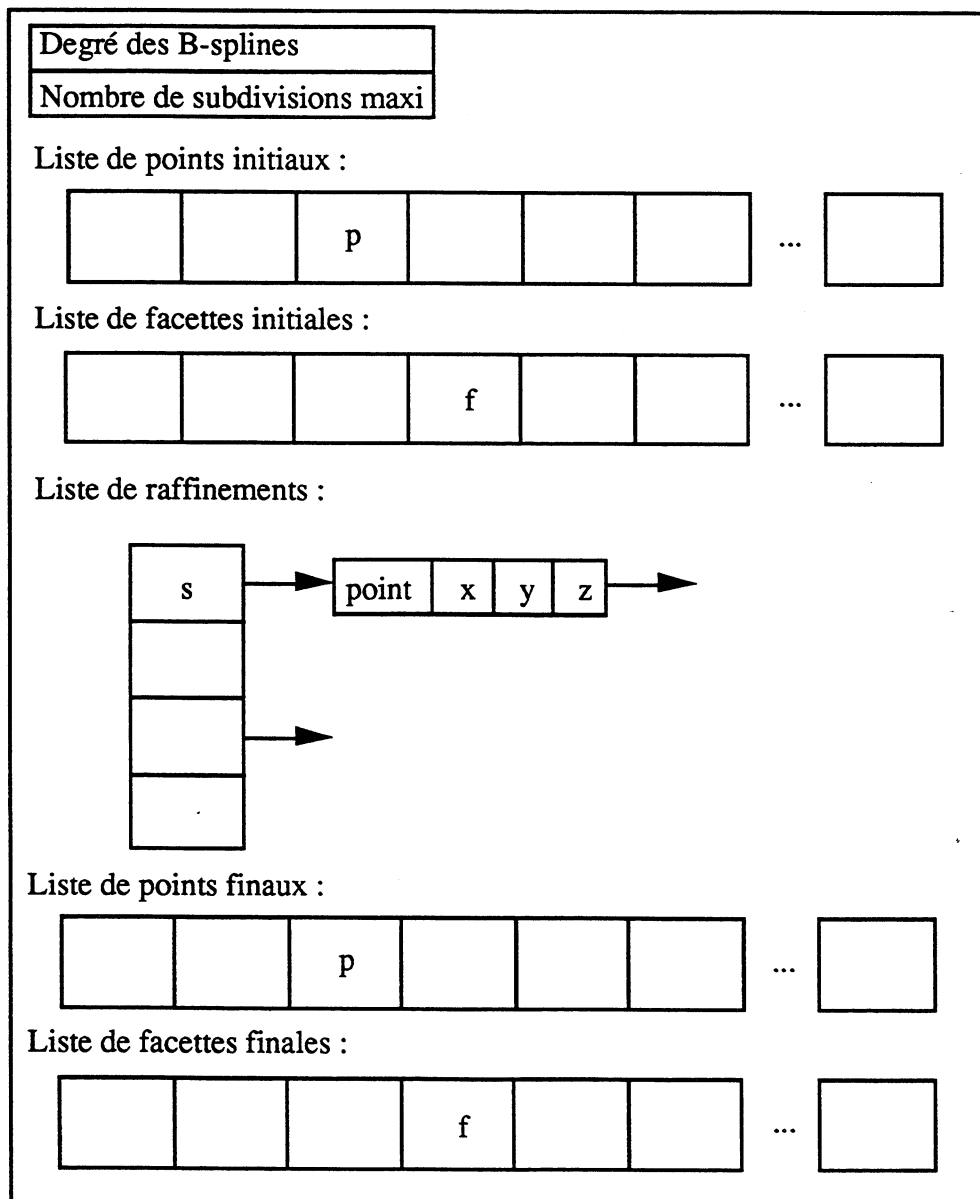


Figure 37 : structure de données d'une surface issue d'un graphe de contrôle

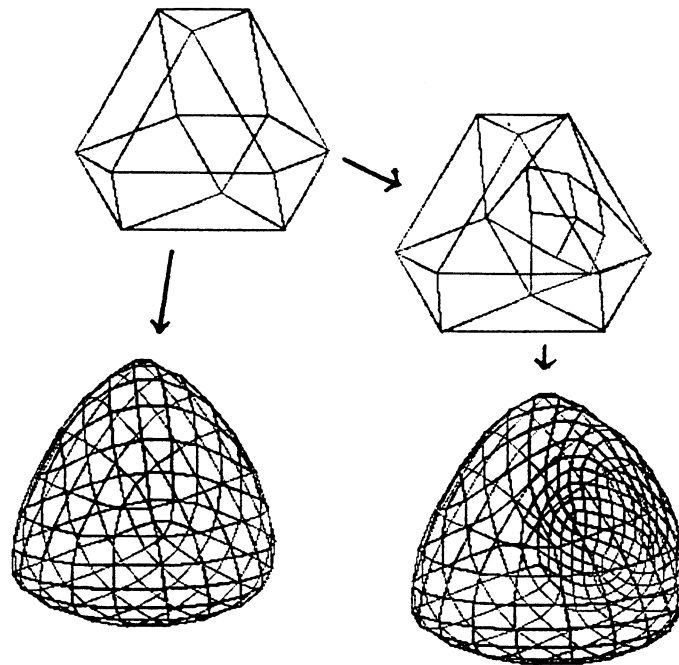


Figure 38 : ajout de points dans un graphe de contrôle

Pour terminer ce paragraphe, nous pouvons dire que la structure de données du graphe de contrôle est plus simple que celle à base de grilles régulières, mais cette simplicité se traduit par des éléments de base (types point et facette) bien plus complexes, de même que les algorithmes associés. Cette représentation est cependant plus agréable à manipuler, une fois les fonctionnalités réalisées.

#### 6.4- La structure scène

A partir des structures surface précédentes, on définit la structure scène, représentant une scène d'objets déformables, issue d'un ou plusieurs objets. La brique de base est donc la surface, et la figure 39 montre le schéma d'implantation. On associe à cette structure les habituelles fonctions d'ajout, suppression et modification d'objets : par exemple, un objet comportant un étranglement crée un deuxième objet, et deux objets s'intersectant se confondent en un seul avec deux surfaces racine.

C'est d'ailleurs au niveau de la structure scène qu'auront lieu les tests d'auto-intersection et d'intersection avec les objets à modéliser. Ces procédures sont détaillées au chapitre 4, qui traite des déformations proprement dites et des effets qu'elles engendrent

Structure scène

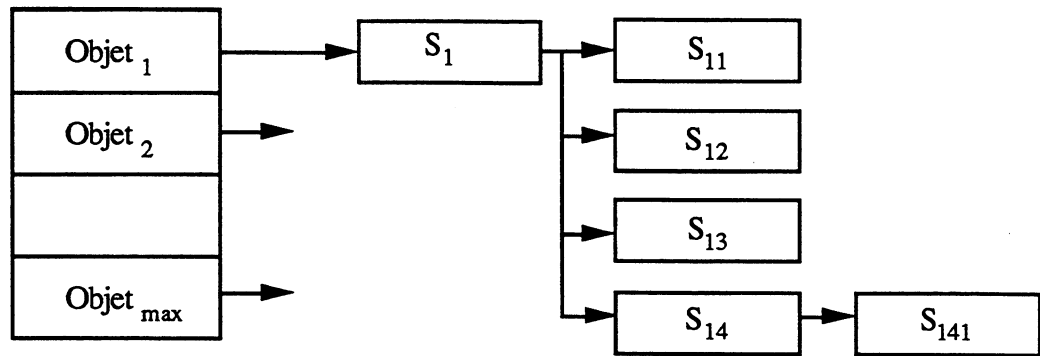


figure 39: structure de scène avec un objet à trou localement sursubdivisé

La structure scène n'est donc qu'une simple table d'adresses pointant sur des surfaces racine (ici  $S_1$ ). Chaque cellule objet contient en outre la position du centre de gravité de l'objet et les paramètres de rotation de l'objet par rapport au repère général de la scène. Ceci permet de déplacer et faire tourner les objets indépendamment et donner à l'utilisateur une vision de chaque objet.

A propos de visualisation, cette structure scène est donc utilisée à travers une interface utilisateur, permettant de manipuler les paramètres d'évaluation, et d'effectuer quelques opérations de rendu sur la scène courante : vue du nuage de points, fil de fer, surface. A l'aide d'une palette de couleurs, on peut visualiser des caractéristiques de normale, courbure, convexité, raffinement...

## 7- Conclusion

On a donc étudié pour la surface active deux représentations à base de produits tensoriels de B-splines, permettant de garder un degré de continuité élevé, et d'obtenir une évaluation facile du graphe de la fonction.

Les raffinements locaux sont de plus pratiques à mettre en oeuvre, puisqu'il suffit de déplacer des points de contrôle à différents stades de l'algorithme de subdivision, utilisé pour l'évaluation ci-dessus.

Les changements de topologie annoncés peuvent s'effectuer en "découpant" les produits tensoriels en sous-produits, puis en effectuant des raccordements. Une représentation de la surface sous forme de liste de carreaux est plus agréable à ce niveau (notamment en termes de continuité), mais en pratique, une surface active ne subira pas plus de deux ou trois changements de ce type (du moins dans une application à caractère médical).

En terme d'algorithmes, la solution à base de carreaux est plus "touffue" que celle utilisant un arrangement de grilles régulières, mais l'utilisation des surfaces devrait

donner l'avantage à cette première solution : en effet, il reste pénible de gérer plusieurs sous-surfaces si leur nombre s'accroît.

De ces deux représentations, il ressort que le modèle du graphe de contrôle sera en moyenne plus adapté aux manipulations que nous effectuerons dans les chapitres suivants.



## **CHAPITRE 4**

### **Dynamique de la surface**

## **1- Introduction**

### **1.1- Préambule**

L'évolution de la surface repose sur le principe de la découverte d'un objet inconnu. C'est une démarche à "tâtons", dont une image intéressante est la prise d'un objet dans l'obscurité. Si on suppose que cet objet flotte dans l'espace et que notre main est autour de l'objet, la découverte de l'objet sera totale quand la main se sera adaptée à ses formes. Ici on va disposer d'une "main" plus universelle au niveau du premier contact puisqu'initialement sphérique, et qui possède beaucoup plus de possibilités d'adaptation (dont celle de changer sa topologie).

### **1.2- Hypothèses initiales**

La surface est active, et aura à respecter un certain nombre de contraintes intrinsèques, ainsi que des contraintes extrinsèques, liées à son environnement. On est donc tout à fait dans le cas des contours actifs "classiques".

Les contraintes intrinsèques vont être liées à la courbure de la surface déformable en chacun des points, et notamment à son signe. Ceci permettra d'une part de fournir un élément moteur, et d'autre part d'éviter des problèmes de repliement.

Les contraintes extrinsèques vont naturellement permettre de lier la surface déformable aux données. On peut rendre les contours à rejoindre attracteurs ou non :

Dans le premier cas, la surface active est attirée par les contours de l'objet, mais si un élément de la surface est à égale distance de deux contours, il y a ambiguïté. Rien n'empêche donc l'apparition de minima locaux. Il faut donc trouver des contraintes intrinsèques pour "dégager" la surface active.

Dans le deuxième cas, la surface se déplace uniquement via des contraintes intrinsèques. Il faut donc la stopper : pour cela, on simule une "solidité" du contour, en empêchant la surface de pénétrer à l'intérieur de l'objet : si dans un mouvement la surface vient à l'intersecter, elle devra reculer. Le fait de rester "à l'extérieur" est de plus une contrainte agréable, puisqu'elle permet de différencier la surface externe de l'objet de ses éventuelles structures internes.

Aussi a-t-on testé un compromis, en empêchant la surface active de traverser des contours, tout en rendant ces derniers attracteurs, via une carte de distance. Précisons tout de suite que l'application de la méthode sur des contours réels (chapitre 5), induit l'apparition ou la disparition de certains éléments de contour, mal détectés : cette carte

n'est donc pas figée, ce qui induit un surcoût de calcul. C'est pourquoi que nous ne présentons cette méthode qu'en Annexe A.

La carte de distance ne résoud cependant pas certains problèmes. En effet, supposons que l'objet possède une cavité interne uniquement accessible par un goulot d'étranglement. A ce point, la carte de distance contient un minimum local, qui va empêcher la surface de pénétrer dans l'objet. Les parties les plus difficiles à rejoindre (généralement des concavités étroites) nécessiteront donc de toutes façons l'utilisation de propriétés intrinsèques adaptées (expansion automatique).

### **1.3- Méthode utilisée**

L'aspect principal de cette méthode consiste à imposer des contraintes internes très fortes à la surface, de façon à obtenir un outil le plus autonome possible, et à réduire d'autant le traitement des données tridimensionnelles. Les contraintes externes vont dans ce cas se réduire à un simple test d'intersection de la surface par rapport aux données (paragraphe 4). On peut aussi les utiliser par l'intermédiaire d'une carte de distance.

La surface étant initialisée en dehors de l'objet, la contrainte interne (paragraphe 3), va minimiser dans un premier temps son aire, puis minimiser son volume. L'approche sera réalisée en tenant compte de la structure hiérarchique de la surface, en déplaçant les points de contrôle d'un niveau de subdivision faible, jusqu'à un niveau élevé. Les changements de topologie seront intégrés à cette étape.

D'un point de vue algorithmique, on utilisera une démarche de type prédiction-corrrection : les contraintes internes prédisent une nouvelle forme, tandis que la correction est assurée par le contrôle de l'intersection données/surface. L'arrêt est conditionné par le blocage de la surface, quand il n'existe plus de points de contrôle susceptibles d'être déplacés, pour une meilleure adaptation de la surface. La figure 1 anticipe légèrement sur le reste du chapitre, et donne une représentation de cet algorithme. On exécute les étapes 1, 2 et 3 du module *Contraintes internes* dans l'ordre, quand la condition de l'étape précédente ne peut être satisfaite.

## **2- Structures de données**

### **2.1- Structure de données "scène à modéliser"**

Pour segmenter un volume de données, on va détecter des indices de contours, puis leur donner une cohérence globale, par le biais d'une surface déformable.



si  $t = 0$ , Mobile = 0-subdivision de la structure de contrôle

**Tant que  $t \leq f$**

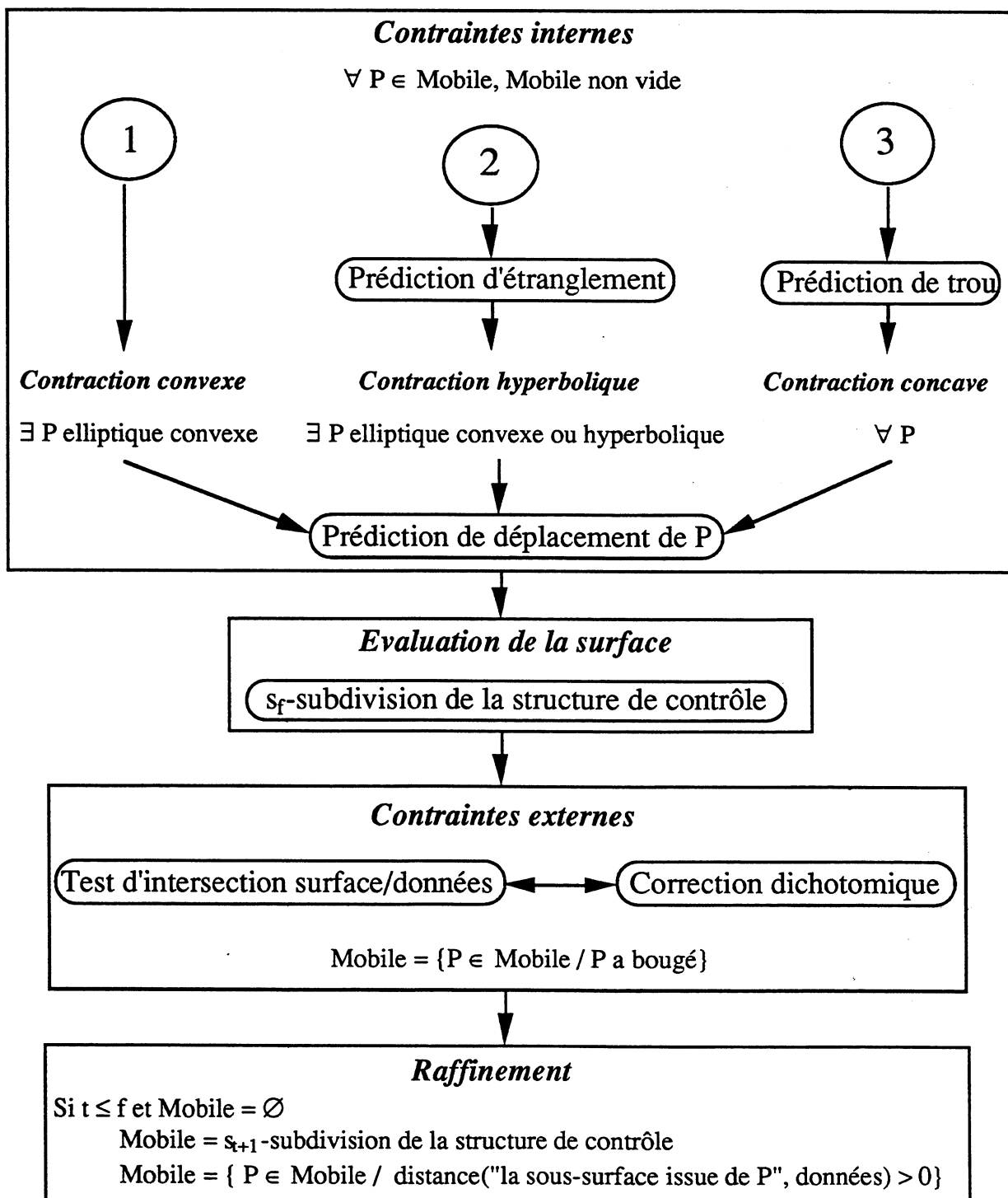


Figure 1 : algorithme synthétique de déformation d'une surface spline subdivisée  $s_f$  fois

La construction de tels indices nécessite l'utilisation de détecteurs de contours [ZUC], comme les lignes de crête du gradient, les passages à zéro du laplacien ou un filtre de Canny-Deriche [CAN]. Cet aspect des choses sera développé de façon plus détaillée dans le chapitre 5, qui traite justement de l'application des surfaces déformables à la segmentation.

Dans un premier temps, on suppose que l'on a une bonne détection, et l'on a seulement à trouver une cohérence globale (au sens d'une surface continue) pour ces contours. L'objet à modéliser se représente alors sous forme d'une bitmap 3D. Mais, pour des raisons de place mémoire, il est difficile d'installer directement cette bitmap. De plus, elle est creuse, puisque les opérateurs utilisés ont en principe détecté des éléments de surface.

On choisit donc une structure de données de type arborescent (figure 2) : un tableau d'entrées bidimensionnel pointant sur des listes de points. Pour un point donné, deux de ses coordonnées (par exemple  $x$  et  $y$ ) désignent la liste correspondante, et il existe une cellule dans cette liste qui contient la troisième coordonnée (ici  $z$ ). Pour cette bitmap, on alloue une cellule uniquement si elle n'existe pas déjà.

Pour convertir une bitmap classique dans ce format, on peut dire qu'un élément  $(x,y,z)$  codé 0 de cette dernière n'a pas de cellule correspondante, alors qu'un élément codé 1 en a une dans la cellule  $z(x,y)$ .

Table de listes

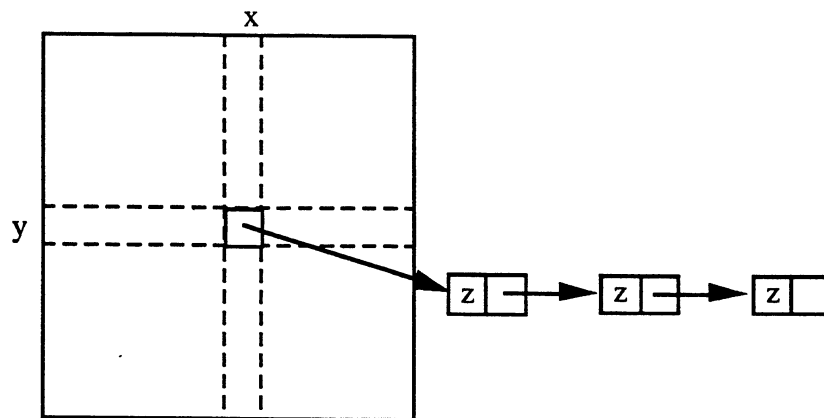


Figure 2 : bitmap 3D arborescente

Pour un calcul de l'efficacité du point de vue de l'encombrement mémoire, on considère qu'un enregistrement utilise un mot mémoire, et un pointeur sur une cellule quatre mots machine (pour un mot de 8 bits). Même en choisissant une telle implantation, on peut montrer qu'il faudrait plus de 20% de remplissage de cette solution arborescente, pour qu'elle soit désavantagée par rapport à une bitmap 3D de 256 éléments d'arête. On peut de plus allouer les cellules par bancs, ce qui réduit considérablement l'encombrement :

pour des bancs de 100 cellules, il faut plus de 75% de remplissage pour atteindre l'encombrement de la bitmap. L'utilisation de champs de bits plutôt que des mots machine conserve des proportions de performance équivalentes, mais la vitesse d'accès est alors réduite.

## **2.2- Structure de données "surface déformable"**

La structure utilisée pour la surface déformable est une application directe des surfaces détaillées au chapitre 3. C'est donc une structure hiérarchique de points de contrôle, que l'on subdivisera jusqu'à une subdivision finale, ou maximale, notée  $s_f$ . On discutera plus loin du choix de  $s_f$  (paragraphe 4.1).

C'est cette structure subdivisée  $s_f$  fois qui permettra de "tracer" l'algorithme, afin de suivre à l'écran l'évolution de la surface. Compte tenu des délais d'affichage, la visualisation de toutes les étapes est fastidieuse, mais on peut par exemple contrôler uniquement la position des points déplacés, et plus rapidement encore, seulement certains stades critiques de l'évolution. Bien sûr, on utilise un rendu adapté selon ce que l'on cherche à voir : trace des raffinements, de la courbure, de la convexité ou des zones de contact.

## **2.3- Structure de données de contrôle**

La surface active évoluera vers le ou les objets à détecter, à partir de sa forme initiale, selon un processus de "raffinement", qui sera décrit ultérieurement (paragraphe 3.5). La forme de la surface dépend uniquement d'un ensemble de "points de contrôle". Ils sont peu nombreux au début de l'évolution, et c'est à partir d'eux que la surface est évaluée, selon l'algorithme de subdivision décrit au chapitre 3. La structure de données "surface déformable" n'est donc que le résultat de cet algorithme à la subdivision  $s_f$ , appliqué à la structure de données de contrôle.

Compte tenu de l'algorithme d'évaluation des points de la surface, on fera évoluer ses points de contrôle par subdivision croissante. On notera  $s_t$  la subdivision courante, ou subdivision de travail, et de même  $s_t$ -polyèdre et  $s_t$ -points, le polyèdre et les points de contrôle de la subdivision  $s_t$ . Pour accroître l'efficacité de l'algorithme, on ne déplacera que certains  $s_t$ -points, choisis par raffinement parmi tous les points de contrôle de la subdivision  $s_t$ . Par abus de langage, on les appellera aussi  $s_t$ -points, sachant que l'algorithme fonctionne quand même si l'on n'utilise pas la technique de raffinement.

La prise en compte d'une nouvelle forme de la surface, construite à partir des contraintes internes, sera simplement répercutée sur la structure de contrôle en plaçant, pour chacun des points, les nouvelles coordonnées à la place des anciennes. Il suffira ensuite d'exécuter l'algorithme d'évaluation jusqu'à la subdivision  $s_f$ , pour se retrouver dans une situation de test par rapport aux contraintes externes.

Dans le cas d'un changement de topologie, il n'y a pas de modification de la position des points de contrôle, mais seulement un réordonnement des grilles de produits tensoriels et des conditions aux bords, ou une modification du graphe de contrôle (selon la modélisation choisie). Ce réarrangement est alors directement effectué dans le calcul de contrainte interne.

### **3- Contraintes internes**

#### **3.1- Stratégie de déformation**

##### **3.1.1- Utilisation de la courbure locale**

Compte tenu de sa position initiale, extérieure à l'objet, la méthode de déformation la plus évidente à laquelle soumettre la surface, consiste à lui appliquer une réduction homothétique, ou encore faire déplacer ses points selon la normale rentrante à la surface. Ce n'est d'ailleurs pas anodin, puisque l'on sait que la pression se répartit uniformément, et applique une force selon cette normale.

Rappelons que nous allons déformer la surface par le biais de ses points de contrôle, et ce par subdivisions croissantes : c'est donc l'ensemble des  $s_t$ -points qui va être soumis à cette avance "normale". Cette pratique rencontre cependant un problème essentiel : si on connaît la direction de déplacement d'un  $s_t$ -point, on ne sait pas sur quelle distance on devra le déplacer, à une itération donnée. Il faut donc décider d'un pas d'avance : s'il est trop petit, le coût de calcul est prohibitif. A l'opposé, s'il est trop grand, la déformation imposée à la surface produit sur ses plus proches voisins une orientation de la normale non dirigée sur l'objet, et fait diverger rapidement l'algorithme en créant des auto-intersections parasites.

La figure 3 donne un exemple bidimensionnel de ce problème : dans les deux cas exposés, on déplace le point  $P_1$  sur la normale rentrante par rapport au polygone de contrôle. Dans le premier cas, on engendre des déplacements pour  $P_2$  et  $P_3$  d'amplitude convenable, alors qu'un déplacement trop important de  $P_1$  dans le second cas engendre des déplacements pour  $P_2$  et  $P_3$  qui induiront une auto-intersection parasite.

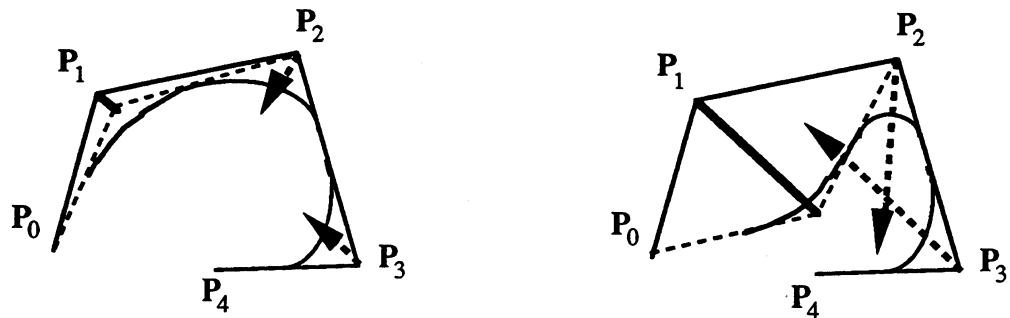


Figure 3 : problème du choix de l'amplitude du déplacement d'un point

De plus, le calcul de cet éventuel pas d'avance devrait être adapté à la monotonie locale (en termes de courbure) de la surface : en effet, les accidents cités ci-dessus seront plus fréquents si la surface comporte beaucoup de changements de courbure. Il est possible de contrôler ses auto-intersections au terme d'une procédure assez lourde, mais même stoppées à l'intérieur de la surface, les déformations ainsi engendrées perturbent le bon fonctionnement de l'algorithme, basé sur la direction de la normale.

Cependant, si on analyse le contre-exemple ci-dessus, on s'aperçoit, que l'erreur constatée dans la figure de droite disparaîtrait, si on empêchait le point  $P_1$  de changer le signe de sa courbure. Dans ce cas, le polygone de contrôle resterait convexe, et l'auto-intersection n'apparaîtrait plus. C'est cette solution que l'on va appliquer dans le cas d'une surface dans l'espace.

On va donc utiliser un critère de déplacement lié à la courbure de la surface en chacun des points. Aussi va-t-on rencontrer trois types de points :

- les points elliptiques convexes.
- les points hyperboliques.
- les points elliptiques concaves.

Les points cylindriques et paraboliques sont classés parmi les points elliptiques. Les points plans ne sont pas déplacés. Par analogie, on nomme les points de contrôle comme elliptiques convexes, hyperboliques ou elliptique concaves selon qu'ils contrôlent un élément de surface convexe, de type selle ou concave. On déduira trois genres de déformations, qui conduiront à déplacer dans un premier temps les points elliptiques convexes, pour produire l'enveloppe convexe de l'objet. On utilisera ensuite les points hyperboliques, qui construiront l'enveloppe localement convexe, définie plus loin. Enfin, le déplacement des points elliptiques concaves fournira la surface externe de l'objet.

### 3.1.2- Choix des points à déplacer

Cette étape combine contraintes internes et externes. En effet, le choix des points à

déplacer est dépendant du degré des B-splines utilisées d'une part, et du fait que la surface intersecte les données d'autre part.

Après avoir classé les points selon leur courbure locale, on cherche à savoir, au sein de chacune de ces classes, quels  $s_f$ -points sont à déplacer. Sachant que les étapes de la boucle de contraction (prédiction, correction) demandent une évaluation et un test de tous les  $s_f$ -points, avec un coût en temps de calcul relativement élevé, il est indispensable de bouger le maximum de points à la fois. Malheureusement, si l'on bouge tous les points d'un  $s_f$ -polyèdre à la fois, et que le carreau ainsi défini heurte l'objet, on est forcé de reculer tous les  $s_f$ -points à la fois puisqu'ils interviennent tous avec un certain facteur sur la partie de la surface en contact. Mais, à cause des corrections du mouvement précédent, des  $s_f$ -points d'un même support peuvent avoir des mouvements contradictoires, et n'arrivent plus à converger. Il faudrait alors réitérer la prédiction de mouvement dans la même direction, sans pourtant être sûr que les points vont progresser, puisque l'avance de l'un peut provoquer le recul du voisin, et vice versa. Une adaptation faisant intervenir les poids des coefficients spline dans le mécanisme de progression s'est révélée inefficace : un point muni d'un faible coefficient recule d'autant plus lentement, et par suite se dégage aussi lentement de l'objet.

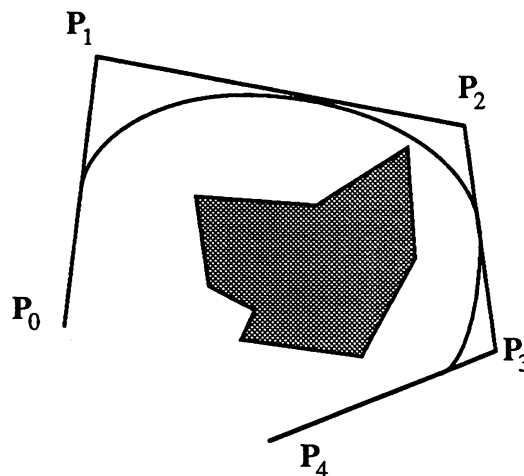


Figure 4 : position initiale du polygone de contrôle

Dans les figures 4, 5 et 6 on illustre ce problème à l'aide d'une spline du second degré. Partant d'une position initiale (figure 4), l'objet à atteindre étant en grisé, on déplace simultanément deux points du même support (figure 5). Les points de la courbe intersectant la frontière de l'objet référencent alors, par le biais de leurs coefficients spline, les points mobiles  $P_1$  et  $P_2$ . D'après ces coefficients, on va donc reculer prioritairement  $P_2$  (figure 6). On va ensuite reculer  $P_1$ . Si on applique un mode d'ajustement dichotomique, et que le recul de tous les points à mi-distance garde encore la courbe dans l'objet, on ne peut pas savoir s'il faut encore reculer ou si certains points

peuvent réavancer : des déplacements contradictoires peuvent avoir lieu, ce qui est d'une part inefficace, et d'autre part, rend l'arrêt de l'algorithme hypothétique.

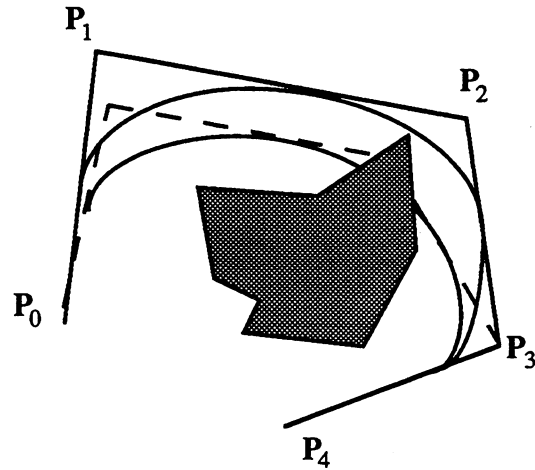


Figure 5 : intersection avec l'objet après déplacement simultané de  $P_1$  et  $P_2$

Il reste possible de déterminer le déplacement de chaque point du  $s_t$ -polyèdre, et ce de manière directe, en résolvant le problème d'interpolation de l'objet par la surface. Le gros inconvénient de cette méthode est qu'il faut connaître les points à interpoler pour résoudre le système. Or c'est justement ce que l'on cherche.

On est donc amené à choisir une famille de points à déplacer de concert, correspondant au déplacement d'un point de contrôle par carreau de surface. Pour terminer une itération d'évolution, il y a en fait autant de sous-itérations qu'il y a de points sur le polyèdre de contrôle d'un carreau de surface. Le choix de cette famille mobile est réalisé par permutation circulaire.

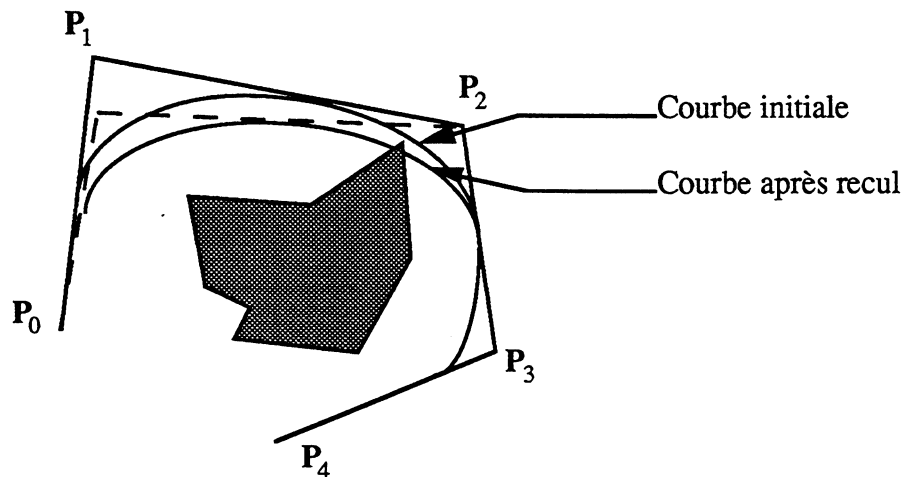


Figure 6 : intersection avec l'objet après recul de  $P_2$

Une solution équivalente consiste à ne déplacer qu'un seul point à la fois sur toute la surface, mais de n'évaluer et tester que le carreau de surface qu'il contrôle.

### 3.1.3- Grilles régulières : traitement des pôles

Un pôle est réalisé pour fermer un produit tensoriel avec de bonnes propriétés de continuité à la jonction. Cette jonction est par définition fermée, mais la grille paramétrique correspondante reste ouverte. Si on prend l'exemple d'une surface sphérique, la grille de contrôle associée est donc de topologie cylindrique, et les pôles se situent sur les bords ouverts.

Voyons quelles contraintes vont être imposées sur cette structure, pour effectuer une déformation homogène (figure 7). Pour un pôle (prenons ici l'exemple du pôle Nord pour faciliter la notation), on dispose en fait de trois points de contrôle mobiles. C'est sur ces trois points que s'exercent les actions visant à contracter la surface. Mais cette structure reste monolithique, pour imposer des contraintes de continuité, et on appellera *surface polaire* l'élément de surface défini à partir de ces trois points. Aussi va-t-on considérer deux opérations :

- une opération de déplacement, qui adapte la position de la surface polaire, dans les limites des conditions de convexité imposées. Un déplacement global du triplet de contrôle assure le mouvement du pôle proprement dit. Ce mouvement correspond à la somme vectorielle pondérée (flèche centrale) des déplacements qu'auraient tous les points de contrôle de la surface polaire.

- une opération de déformation, qui adapte la forme de cet élément de surface. Ceci correspond donc aux déplacements isolés des trois points (flèches en plein), assurant par là l'orientation et les modifications de forme de la surface polaire. L'adaptation de forme peut amener à une contraction du triplet sur lui-même, c'est-à-dire sur le pôle (flèches en pointillés), sans pourtant avoir rencontré un obstacle. Pour éviter la perte de la continuité géométrique au pôle, on les oblige (dans un premier temps du moins) à respecter une distance minimum.

Cette obligation, spécifique aux pôles, provient du fait que l'on a explicitement mis en place une structure de conservation de la continuité : la méthode du triplet minimum permet de retarder une éventuelle perte de cette continuité, tant que la surface n'est pas au contact de l'objet.

Cette contrainte de distance minimum au sein du triplet est donc supprimée dès que la surface polaire a enregistré un contact, pour l'adapter à la continuité des données. Ce test est effectué pour chaque niveau de subdivision, si la subdivision précédente n'a pas déjà concentré le triplet de contrôle. Ceci permet de garder les conditions de continuité initiales pour tous les niveaux de raffinement, tant que la surface polaire n'a pas déjà été adaptée.



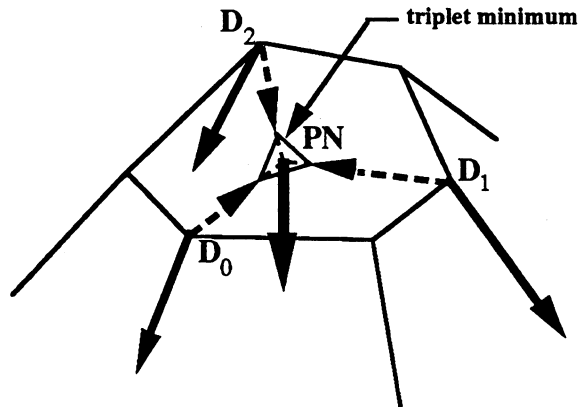


Figure 7 : contraintes de déplacement au pôle

Le problème du choix du point de contrôle à déplacer à une itération de l'évolution est conditionné par la détermination des points impliqués dans un contact. Compte tenu des conditions de continuité imposées au pôle, chacun des points du triplet de contrôle a une action sur la surface polaire, aussi aucun point des carreaux voisins ne pourra-t-il être bougé en même temps. Il faut donc rajouter une famille distincte de points de contrôle, à déplacer séparément au cours de la boucle d'évolution. Cette famille représente tous les triplets des pôles présents sur la surface.

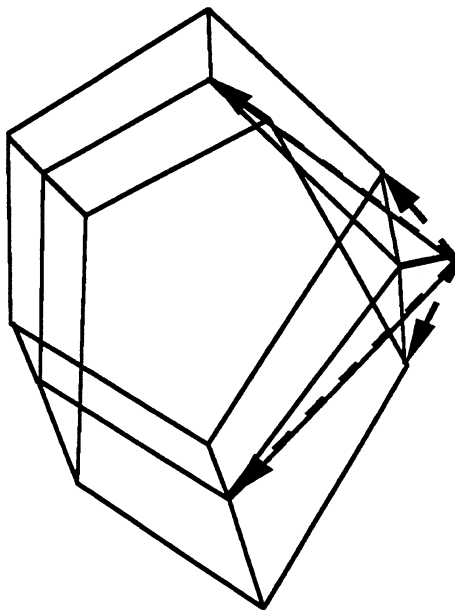
### 3.2- Contraction convexe

#### 3.2.1- Déplacement des points elliptiques convexes

On utilise, dans une première étape, les points elliptiques convexes. Rappelons que la surface est initialisée au départ hors de l'objet. Pour simplifier cette initialisation, on choisit une surface convexe, et les points de contrôle initiaux sont donc tous elliptiques convexes. On cherche alors à les déplacer de manière à réduire l'aire de la surface. Cette première étape est bien représentée par l'image d'un ballon (de volume nul au repos) qui se dégonfle. Compte tenu de l'élasticité d'un tel modèle, chaque point de sa surface est attiré par son voisinage de manière à diminuer l'aire du ballon. Ce faisant, le ballon se colle sur l'objet et décrit son enveloppe convexe. On applique donc cette stratégie sur le  $s_T$ -polyèdre, où chaque  $s_T$ -point elliptique convexe est déplacé en direction du barycentre de ses voisins (figure 8). On restreint ainsi le  $s_T$ -polyèdre, permettant de simuler une tension sur de grandes zones à la fois. On vérifie de plus, que la position atteinte par le  $s_T$ -point déplacé conserve la convexité locale.

Cet algorithme converge donc vers un  $s_T$ -polyèdre convexe, mais à l'infini seulement, puisque les déplacements prédits tendent vers zéro. A partir d'une certaine itération, compte tenu de la discrétisation des données, le  $s_T$ -polyèdre reste immobile, et n'a pas

rejoint la position finale théorique. A ce stade, on pallie à cet inconvénient en effectuant une *itération normale* (au sens du paragraphe 3.1.1), qui suit une progression normale, munie d'un pas très fin (en pratique 2 voxels). Certains  $s_t$ -points elliptiques convexes peuvent ainsi devenir hyperboliques ou elliptiques concaves.

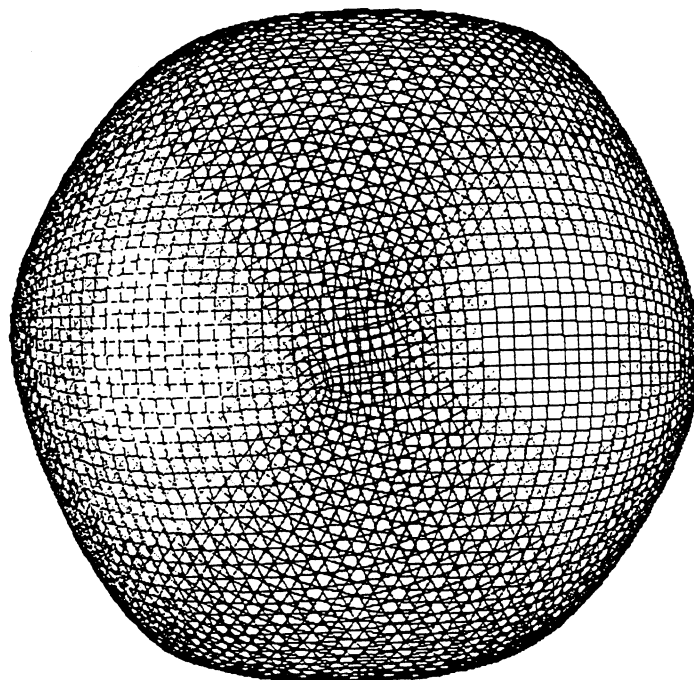


*Figure 8 : déplacement "barycentrique" des points de contrôle elliptiques convexes*

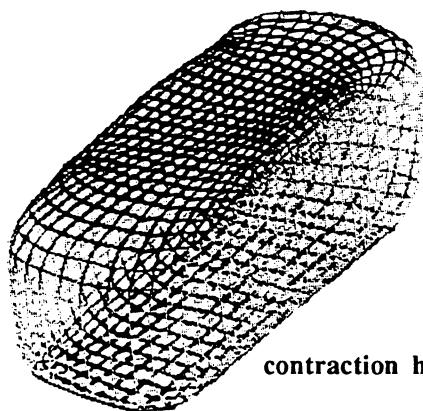
Si des  $s_t$ -points ont progressé lors de l'itération normale, on réalise alors une nouvelle étape de contraction convexe. Notons qu'à l'issue de la contraction convexe (figure 9), il existera, si l'objet n'est pas convexe, un certain nombre de points légèrement (ils présentent une courbure faible) hyperboliques ou elliptiques concaves, qui vont permettre d'initier les deux prochains types de contraction.

### **3.2.2- Adaptation de la continuité géométrique**

Au delà d'une progression "normale" en direction du barycentre de ses voisins, un  $s_t$ -point peut se déplacer uniquement dans la direction d'un de ses voisins, de manière "tangentielle". Ceci permet, au moyen de regroupement de points, la parfaite adaptation au degré de continuité géométrique de l'objet. En effet, si deux points d'un polyèdre de contrôle sont confondus, les caractéristiques de continuité analytique restent les mêmes, mais on perd un degré de continuité géométrique.

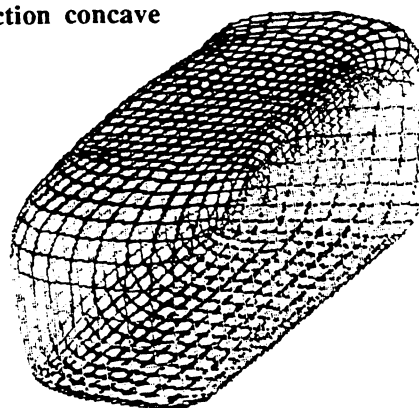


**contraction convexe**



**contraction hyperbolique**

**contraction concave**



*Figure 9: adaptation à une forme convexe*

Par exemple (figure 10, 11) un angle droit peut parfaitement être modélisé par une courbe spline du second degré, si deux de ses points de contrôle sont situés sur cet angle droit (figure 11). Ainsi, le déplacement d'un  $s_t$ -point vers un seul de ses voisins rapproche ces deux points (au contraire d'un déplacement vers le barycentre des voisins, qui "équilibre" la répartition spatiale) : la présence d'une zone de continuité moindre que celle de la surface utilisée va permettre à certains points de se regrouper pour adapter la surface, laquelle perdra alors un ou plusieurs degrés de continuité géométrique.

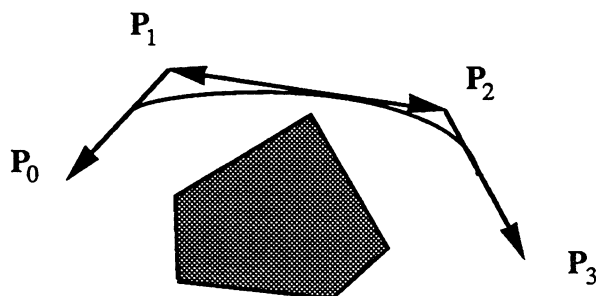


Figure 10 : position avant l'étape de progression tangentielle

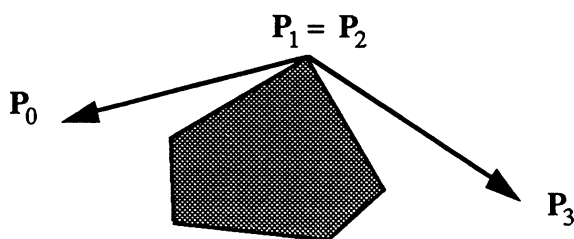


Figure 11 : position finale des points  $P_1$  et  $P_2$  sur l'angle droit

Dans l'étape de contraction, le déplacement de chaque point devient le suivant : si un  $s_t$ -point elliptique convexe n'atteint pas le barycentre de ses voisins, en restant elliptique convexe, il va tenter de se déplacer en direction de chacun de ses voisins, avec la même condition de convexité locale. S'il peut en rejoindre plusieurs, on le placera sur le barycentre de ces points. Aussi, l'ordre dans lequel il cherche à rejoindre ses voisins n'a pas d'importance.

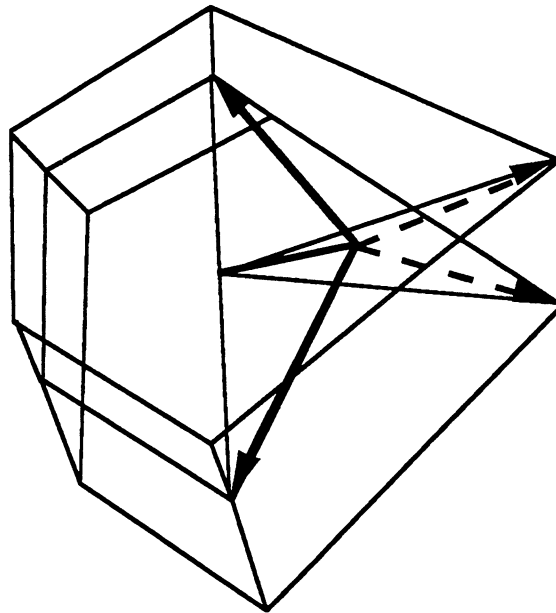
### 3.3- Contraction hyperbolique

#### 3.3.1- Elasticité directionnelle aux points hyperboliques

On définit l'*enveloppe localement convexe* comme étant la surface de plus petit volume englobant l'objet et ne présentant aucune concavité stricte. Elle est constituée d'un nombre minimal de composantes connexes convexes (par exemple la surface d'un objet constitué de deux sphères d'intersection non nulle est différente de son enveloppe convexe, présente au minimum deux composantes convexes, mais n'a pas de

concavités). La surface ainsi produite comporte des zones convexes, ainsi que des selles, où la courbure n'a pas le même signe dans toutes les directions. C'est donc une étape de plus que l'enveloppe convexe précédente, qui va donner une forme plus affinée des objets, et surtout qui va permettre de résoudre les problèmes d'objets multiples (puisque cette enveloppe tend à "s'infiltrer" entre les enveloppes convexes des objets situés à l'intérieur de la surface déformable). Pour parvenir à ce type d'enveloppe, nous allons déplacer les points hyperboliques. On ne garantit cependant pas la minimalité du nombre de composante connexes convexes constituant l'enveloppe obtenue, ni son unicité, mais les expériences ont été jusqu'à présent satisfaisantes. De même, la définition proprement dite ne présente qu'un caractère didactique, puisque l'on n'utilisera pas d'autre propriété que celle de ne pas présenter de concavités strictes.

Pour construire cette enveloppe localement convexe, on va utiliser la même méthode que précédemment dans le modèle du ballon, mais en modifiant les contraintes d'élasticité, par suppression des tensions rappelant la surface sur son enveloppe convexe. Les zones de la surface déformable en rapport avec des zones strictement convexes ou concaves de l'objet, vont rester inchangées dans cette étape. Une bonne image consiste à utiliser une main comme objet à atteindre. Une fois l'enveloppe convexe atteinte, on permet à la surface de s'infiltrer entre les doigts, et le résultat ressemble à un gant.



*Figure 12 : choix des directions de déplacement pour un point de contrôle hyperbolique*

Le déplacement des points de contrôle pour une subdivision  $s_t$  fonctionne strictement de la même façon que précédemment, avec une nuance dans le choix des directions (figure 12). Chaque point elliptique convexe tentait de se diriger vers ses quatre voisins : un point hyperbolique ne se dirige vers un de ses voisins, que si ce déplacement n'augmente pas le volume inclus dans la surface. Un bon critère revient à tester le produit scalaire entre la normale sortante de la surface au point considéré, et la direction de déplacement

(flèches). S'il est positif (flèches en pointillés), le déplacement tend à accroître le volume, et cette direction de déplacement n'est pas utilisée (dans ce cas, on pourrait inverser le sens du déplacement, mais rien ne prouve la stabilité d'un tel choix).

Ce calcul se faisant au niveau de la subdivision  $s_t$ , l'évaluation des éléments nécessaires se fait aussi à ce niveau. La normale est donc la normale au  $s_t$ -polyèdre, calculée discrètement. L'intérieur et l'extérieur de la surface sont déterminés en un point P, grâce au signe du produit vectoriel des vecteurs liant P à ses voisins sur le  $s_t$ -polyèdre, pris deux à deux, dans un certain ordre. D'autre part, calculer la normale à partir de l'équation de la surface est certes possible, mais il faut savoir en quel point de la surface, et les propriétés qui nous intéressent sont celles de la grille à la subdivision  $s_t$ . C'est tout le but de la démarche hiérarchique, et on peut montrer (par récurrence) que le  $s_t$ -polyèdre et la surface (évaluée à la subdivision  $s_f$ ) ont mêmes propriétés, au point de vue de la convexité locale.

La stratégie proposée consiste à prendre pour point de départ l'enveloppe convexe de l'objet, obtenue après l'étape de contraction convexe. Sur cette enveloppe convexe, les points de contrôle se situent dans des conditions de voisinage impliquant un équilibre des tensions pour un critère de convexité : compte tenu des données, certains n'ont donc pas pu dépasser cette enveloppe convexe (en fait ils sont hyperboliques). Maintenant on supprime les tensions qui rappellent ces points sur cette enveloppe, et ils vont pouvoir progresser à nouveau vers l'objet (figure 13).

### 3.3.2- Auto intersection

Suite à ce nouveau mode de déplacement, une scène constituée de plusieurs objets non connexes va donner lieu à un étranglement. En effet, il existe un ensemble de points, partant de l'enveloppe convexe, qu'au moins une direction de déplacement va entraîner dans l'espace situé entre ces objets. Comme il n'y a pas de données entre les objets, cette dernière direction va pouvoir être suivie, et cet ensemble de points va se contracter sur lui-même : la tension étant équivalente, il converge vers un point, le point d'étranglement. Il peut bien sûr y avoir plusieurs étranglements simultanés.

La détection d'auto-intersection peut donc être réalisée en fin de cette étape de contraction hyperbolique, et va donner lieu à une découpe de la surface en deux ou plusieurs surfaces. Ceci crée un certain nombre de points elliptiques convexes, qui ne sont pas en contact avec un objet. Dans ce cas, on retrouve une situation similaire à la position initiale de la surface au début de l'algorithme, et il faut relancer une étape de contraction convexe sur chacune des sous-surfaces ainsi engendrées, avant de revenir à l'étape de contraction hyperbolique interrompue (figure 14).

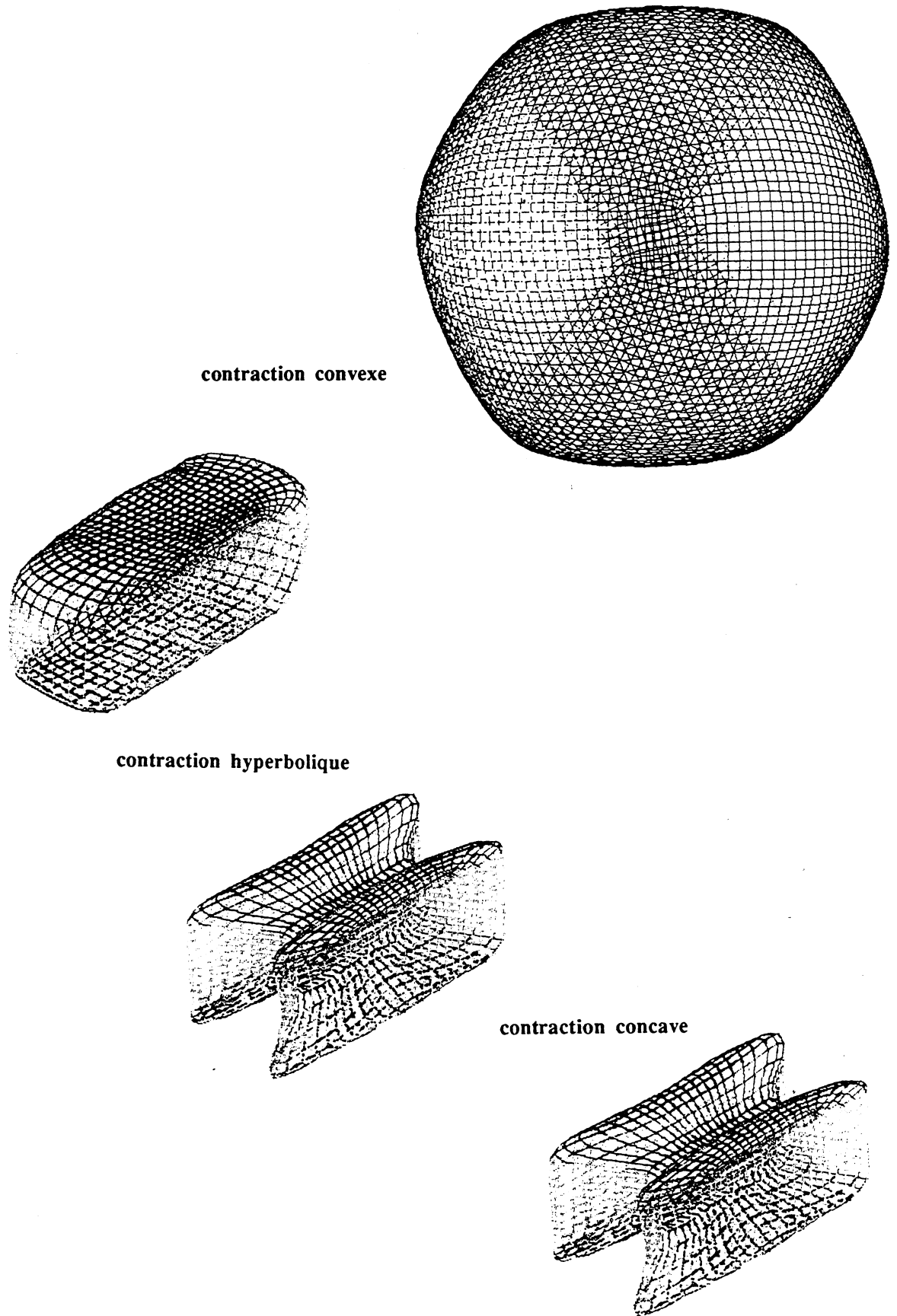


Figure 13 : adaptation à une forme présentant une selle

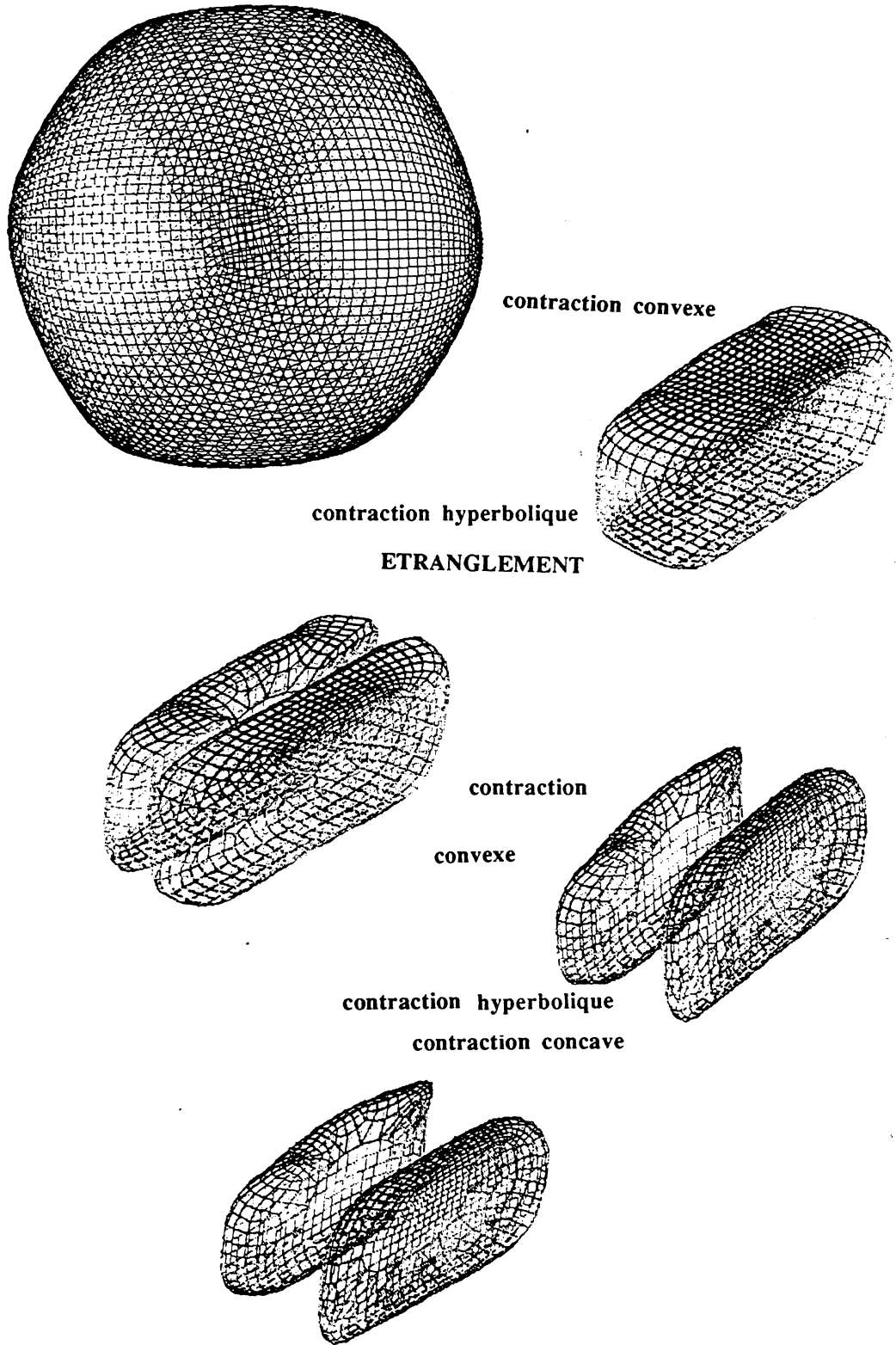


Figure 14: adaptation à une forme "double"



### 3.3.3- Détection d'étranglement

L'étranglement correspond, comme on l'a vu au chapitre 3, à un repliement spatial d'un ensemble de points formant une courbe fermée dans le repère lié à la grille de contrôle (ou un chemin fermé dans un graphe de contrôle). Deux solutions sont possibles pour gérer ce phénomène. La première consiste à séparer les zones de la surface séparées par l'étranglement, après l'avoir détecté. La seconde teste un étranglement possible, avant qu'il ne se forme.

La première solution utilise donc une procédure de détection de l'étranglement. Présentons deux techniques différentes : l'une est analytique et utilise une propriété de l'étranglement pour déterminer ses points. L'autre est discrète et repère des points confondus, puis les trie *a posteriori* selon une propriété de courbure.

- Pour la première technique, on remarque qu'un carreau de B-spline sera impliqué dans un étranglement s'il contient un ensemble de points de même lieu géométrique, le point d'étranglement. Il faut cependant résoudre ce problème pour tous les carreaux de la surface, et tenir compte de carreaux dégénérés (par exemple des points de contrôle ayant même position spatiale, réduisant ainsi la continuité géométrique). La prise en compte de ces derniers revient alors à tester la grille de contrôle elle-même.

- La deuxième technique détecte tous les  $s_f$ -points de la grille ayant même position spatiale, puis vérifie que ces points forment, sur la surface, la frontière de deux (ou plusieurs) domaines.

Table de listes

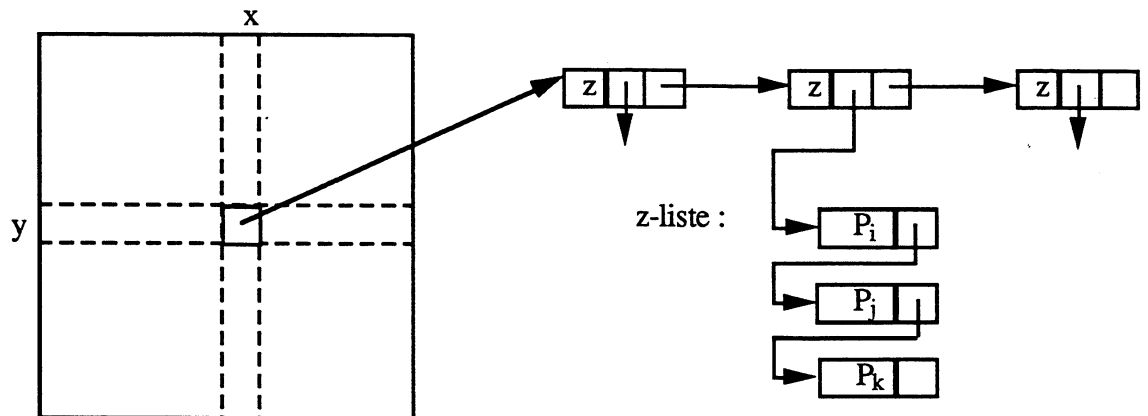


Figure 15 : structure d'auto-test

La détection de points confondus peut se faire de manière similaire à la détection d'intersection objet/surface. On utilise une structure de données tridimensionnelle arborescente (figure 15), dont chaque entrée contient une liste de points (appelons-la z-liste). Il suffit de parcourir le  $s_f$ -polyèdre, et de remplir cette structure.

Une z-liste contenant plus d'un point n'est pas forcément un étranglement : dans la grande majorité des cas, c'est simplement un effet de la discrétisation de l'évaluation de la surface. Aussi utilise-t-on la courbure locale de ces points. D'après leur position, ils ont pour propriété d'être hyperboliques. Par suite, un étranglement est formé par une z-liste dont tous les points sont hyperboliques. Il faut alors retrouver les  $s_t$ -parents hyperboliques de ces points, pour construire les sous-surfaces.

La seconde solution pour gérer un étranglement consiste en sa prédiction, en utilisant directement les  $s_t$ -points hyperboliques qui se sont déplacés, et qui séparent la surface en deux domaines. On testera la surface formée, en simulant un étranglement à partir de ces points, qui sera conservé s'il n'intersecte pas les données.

Pour vérifier qu'une famille de  $s_t$ -points est la frontière de deux (ou plusieurs) domaines, il suffit de colorier les points du  $s_t$ -polyèdre en respectant la condition suivante : on ne peut colorier un point que si son voisin l'est déjà, et qu'il n'appartient pas à la frontière. Après avoir colorié un point quelconque (n'appartenant pas à la frontière) du polyèdre, et itéré en respectant la condition ci-dessus, deux cas se présentent : on a colorié tous les points (hormis la frontière), ou non. Dans le premier cas, la frontière est fautive, et il n'y a pas d'étranglement. Dans le second cas, la famille de  $s_t$ -points hyperboliques sépare bien la surface en deux (ou plusieurs) domaines. On conserve, parmi les points frontière, ceux qui bordent la région coloriée, et on teste les sous-surfaces issues de cet étranglement.

L'avantage décisif de cette approche prédictive est que l'on peut effectuer ce test dès le début de la contraction hyperbolique, ce qui permet de gagner du temps (il n'est pas nécessaire d'attendre que tous les points frontière se concentrent), puis de manière aléatoire. D'ailleurs, si un étranglement doit avoir lieu, pour une subdivision  $s_t$ , elle se fera vraisemblablement dès le départ, un cas plus complexe pouvant être résolu à une subdivision supérieure.

### 3.4- Contraction concave

#### 3.4.1- Adaptation aux concavités

La progression précédente a amené la surface déformable sur une enveloppe de l'objet, ne possédant aucune concavité locale (définie par un convexe appartenant au complémentaire

de l'objet dans son enveloppe localement convexe) : il reste donc à traiter celles-ci, ainsi que les trous éventuels de l'objet. Pour cela on cherche à utiliser une démarche opposée à la première étape, modélisant une tension surfacique. Chaque point est repoussé par ses voisins, dans une étape de "répulsion surfacique", visant à réduire le volume contenu par la surface.

Pour un point elliptique concave, ceci revient à utiliser strictement les mêmes directions de déplacement que pour la convergence convexe, mais dans le sens opposé (figure 16). C'est donc une solution de déplacement tout à fait divergente, mais qui est contrôlée grâce à la situation particulière de la surface.

La surface, à cause de ses déformations précédentes, "oblitére" entièrement les concavités de l'objet. En effet, une concavité est bordée de points qui ne sont pas elliptiques concaves, et le propre de la méthode utilisée est d'avoir déjà rejoint ces points-là. On cherche donc à minimiser le volume qu'elle contient, en lui faisant subir une expansion à l'intérieur des concavités. Le déplacement de ses points sera donc localement divergent, et il ne peut pas se produire d'auto-intersections parasites, au sein d'une même concavité (Nous ne donnons pas de preuve formelle à cette assertion, mais les expériences tendent à prouver la robustesse du processus). On utilise la somme vectorielle moyennée des déplacements de la répulsion, pour retrouver un déplacement selon la normale, mais dont l'amplitude est fixée (figure 18). La figure 17 montre ainsi une convergence sur un parallélépipède arrondi présentant deux concavités.

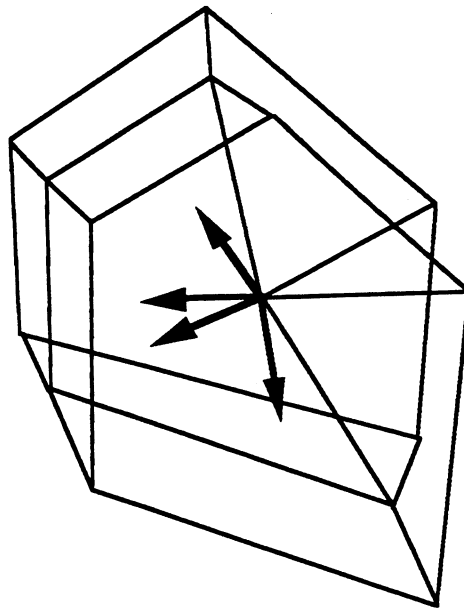
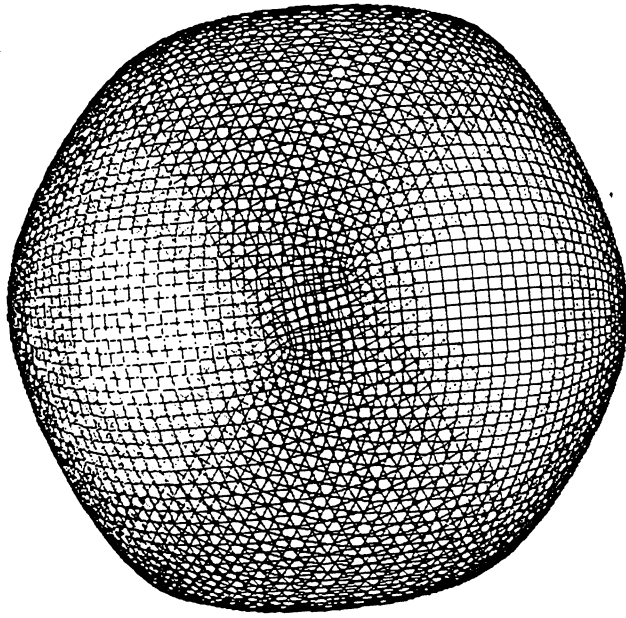
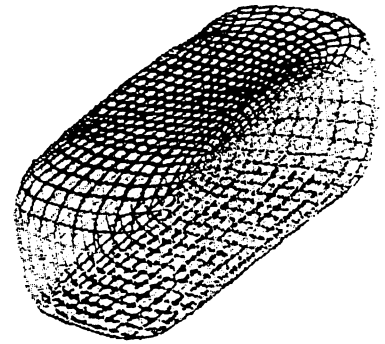


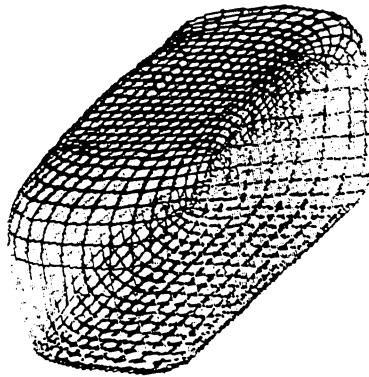
Figure 16 : répulsion d'un point par ses voisins



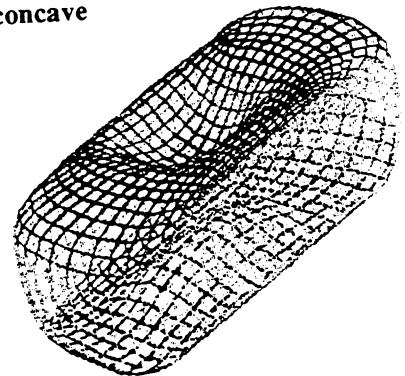
**contraction convexe**



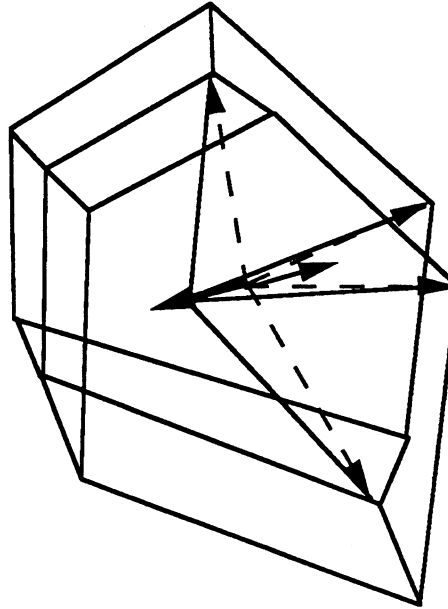
**contraction hyperbolique**



**contraction concave**



*Figure 18 : adaptation à une forme présentant des concavités*



*Figure 17 : direction de déplacement des concavités selon la normale rentrante*

Cette solution de déplacement va donc amener la surface au contact de l'objet, mais ne s'adapte pas à ses propriétés de continuité géométrique, comme le faisaient les deux premières étapes. Après avoir utilisé la somme pondérée des vecteurs répulsion, pour placer la surface au fond des concavités, on utilise directement ces vecteurs tour à tour : les points ayant tendance à se regrouper dans les zones de continuité géométrique faible, on retrouve alors cette propriété d'adaptation.

### **3.4.2- Auto intersection**

La progression de la surface dans les concavités n'est malheureusement pas aussi simple qu'on le voudrait. En effet, toute concavité peut cacher un trou. Dès que l'on aborde l'étape de convergence concave, il faudrait donc tester, pour tout déplacement d'un point elliptique concave, l'apparition d'une auto-intersection, suite à l'existence d'un trou dans l'objet recherché. Plusieurs trous peuvent bien sûr apparaître en même temps.

Il faut alors générer ce trou dans la structure de la surface, et l'on obtient, au niveau du trou, un certain nombre de points de contrôle, de type hyperbolique. On considère tout simplement cette situation comme une nouvelle initialisation de la surface, mais dont tous les points elliptiques convexes sont déjà en position : on réapplique alors la méthode, en partant de l'étape de contraction hyperbolique, qui va se charger "d'élargir" le trou, puis continuer l'étape concave, interrompue par le changement de topologie (figure 19).

### 3.4.3- Détection de trou

La détection d'auto-intersection de type "trou" sur la surface peut se réaliser de deux manières différentes. La première consiste à "percer" la surface après contact, et donc de chercher exactement où la surface s'auto-intersecte. La seconde consiste à prévoir cette auto-intersection, et donc à "percer" avant le contact.

Pour la recherche d'auto-intersection, voyons une possibilité de détection analytique, et une autre, discrète :

- chaque carreau de B-spline composant la surface a une équation paramétrique : il suffit d'écrire que deux carreaux ont une intersection, et de résoudre le système (3 équations et 4 inconnues pour une bicubique par exemple) correspondant, pour les carreaux pris deux à deux, afin de trouver ainsi les points en contact. C'est cependant lourd en termes de performances, même si on restreint le test aux carreaux s'étant déplacés (ils correspondent tous à une concavité s'ils se déplacent dans cette étape).

- la solution discrète a l'avantage d'être rapide puisqu'elle fournit les points candidats en un temps équivalent à un simple parcours de la surface : on remplit une structure tridimensionnelle (contenant pour chaque entrée une liste de  $s_f$ -points ayant même coordonnées spatiales) chaque fois que l'on a déplacé un point de contrôle elliptique concave. Le critère est que les points en contact doivent appartenir à des carreaux de base différents, aussi une entrée de cette structure, correspondant une auto-intersection, ne peut-elle pas contenir des points paramétriquement voisins. La technique consiste donc à vérifier si les points d'une entrée décrivent un ensemble disconnexe sur la grille de contrôle. Pour réaliser le trou, on fournit alors un  $s_t$ -point parmi les  $s_t$ -parents de chaque famille disconnexe composant l'entrée candidate.

Cette recherche d'auto-intersection reste cependant lourde, puisqu'il faut l'exécuter systématiquement dès qu'un point elliptique concave est déplacé. Aussi va-t-on préférer une technique prévisionnelle.

La seconde manière de gérer la création d'un trou va se faire en prédisant un trou entre deux points elliptiques concaves, et en testant la surface ainsi formée : si elle intersecte les données, la prédiction est abandonnée, sinon la transformation devient effective.

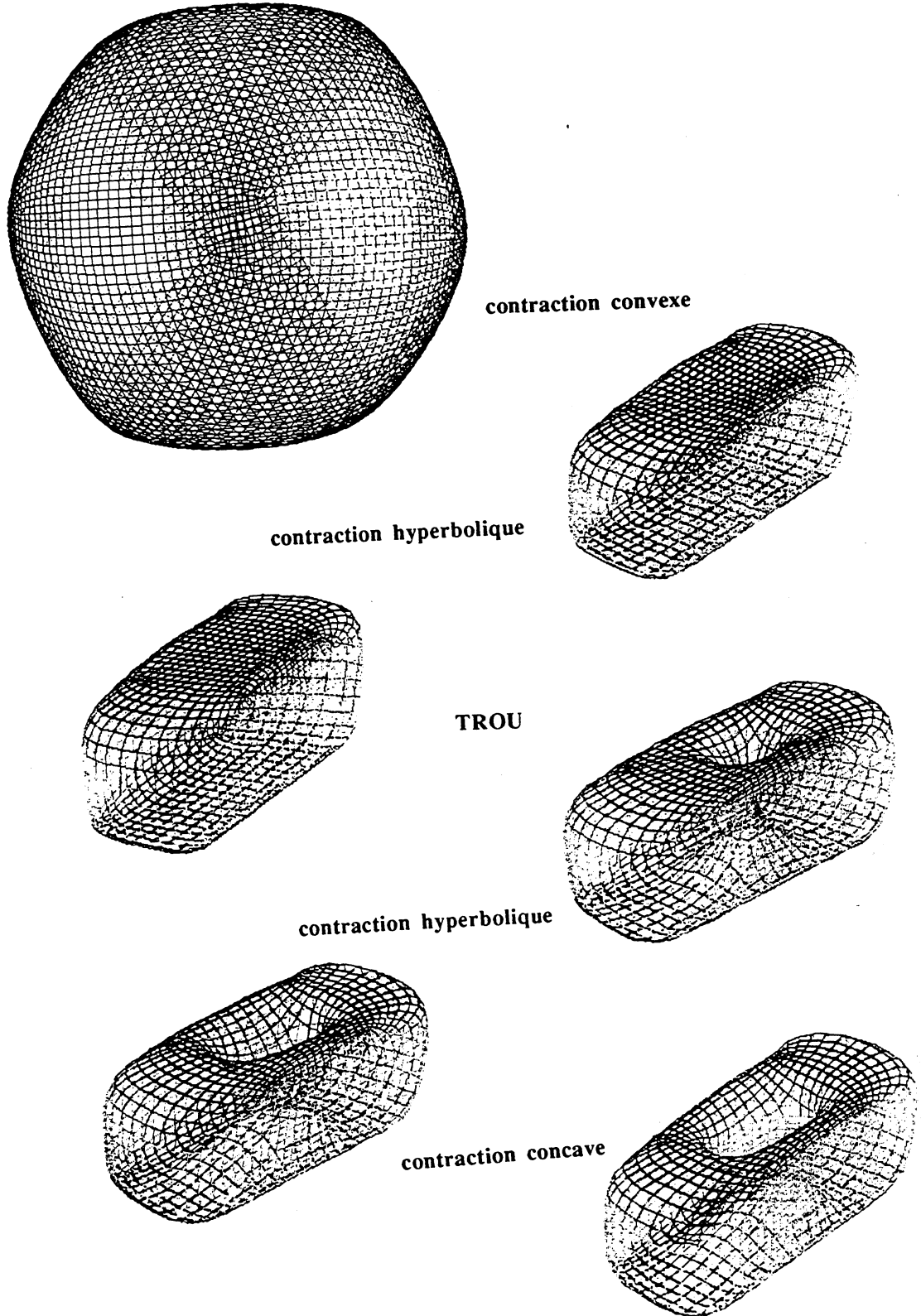


Figure 19 : Adaptation à une forme torique

Les deux points choisis sont des  $s_t$ -points, ce qui limite leur nombre, et rend leur choix plus approprié que dans la solution précédente. Ils doivent de plus s'être déplacés, et avoir eu un déplacement contraire, dans une direction sensiblement équivalente à la droite qui passe par leurs positions respectives (on admet un angle de 30 degré autour de cette droite). On teste donc tous les couples de  $s_t$ -points satisfaisant ces conditions. On effectue ce test dès le début de l'étape de contraction concave, et ensuite par intervalles de sécurité non nécessairement réguliers. En effet, s'il existe un trou à réaliser à la subdivision  $s_t$ , il sera vraisemblablement réalisé dès le départ, un trou "tortueux" réclamant une subdivision supérieure. L'intérêt de cette méthode consiste donc en un nombre de tests relativement réduit, et une plus grande rapidité de convergence, puisque si un trou est réalisé, il n'est pas utile d'attendre que la surface s'auto-intersecte véritablement.

### **3.5- Raffinements**

#### **3.5.1- Généralités**

Les raffinements de la surface déformable interviennent pour donner plus de souplesse à celle-ci, et permettent une adaptation parfaite sur les données. Rappelons que le raffinement d'un  $s_t$ -point (chapitre 3, paragraphe 4) correspond à donner une "autonomie de déplacement" à ce point, sa position étant jusqu'alors déterminée par l'algorithme de subdivision ( $s_t$  subdivisions).

Le nombre de raffinements est de toute façon limité par le nombre de points finaux des grilles de contrôle. Les tests d'intersection se faisant à ce stade, un raffinement, réalisé sur un point de subdivision finale, conduit à considérer la surface comme polyédrique, puisqu'il n'y a plus de subdivision possible. Il faut donc utiliser une surface ayant un nombre final de subdivisions important, mais en gardant à l'esprit qu'un fort taux de subdivision ralentira de manière sensible la vitesse d'évaluation.

Toujours pour des raisons de performance, il est utile d'avoir peu de points de contrôle sur la grille initiale, pour deux raisons :

- si la surface n'entre pas tout de suite en contact avec l'objet, il va falloir déplacer un grand nombre de fois les points mobiles pour obtenir la localisation grossière de l'objet, et par là-même sa première approximation. La performance en termes de vitesse d'évolution est alors d'autant plus faible que le nombre de points mobiles initiaux est élevé.

- le déplacement choisi étant du type tension surfacique, et la prédiction se faisant à la subdivision de travail, un grand nombre de points sur la grille initiale diminue



l'amplitude des déplacements et par là même la vitesse de convergence.

Ces constatations sont d'autant plus vraies que l'objet à découvrir est petit devant la surface initiale. Notons que la technique de raffinement apporte un facteur de vitesse, puisqu'elle permet de bouger uniquement les points utiles, et un facteur de représentation, permettant d'appréhender la forme de l'objet de manière hiérarchique. L'algorithme d'évolution est cependant en mesure de travailler directement avec tous les points d'une subdivision (voire la subdivision finale dans le cadre d'une approche polyédrique) et de converger vers le même objet. Dans ce cas, la vitesse de convergence est bien sûr bien plus lente (notamment au début, lorsqu'on est loin de l'objet).

### 3.5.2- Raffinement par subdivisions successives

Sachant que la boucle principale de l'algorithme d'évolution est en mesure de faire converger une grille de contrôle donnée, cette boucle sera appelée successivement pour chaque subdivision de travail. Aussi, lorsqu'une subdivision de travail  $s_{t-1}$  donnée a vu tous ses points mobiles arriver à destination, on a obtenu une nouvelle approximation de l'objet recherché. On décide alors de modifier la subdivision de travail suivante  $s_t$  (plus fine). Il faut donc utiliser une technique de raffinement appropriée pour rendre mobiles certains de ces points de contrôle.

Le critère de raffinement utilisé est le suivant : un  $s_t$ -point est susceptible d'être raffiné si moins de deux  $s_f$ -points de l'élément de surface qui dépend du  $s_t$ -point sont au contact immédiat de l'objet. En effet, si aucun point de cet élément de surface n'est au voisinage de l'objet, elle peut encore se déplacer globalement plus loin. Si un ou deux points sont au contact, l'élément de surface peut encore effectuer une rotation.

On va tester la position de chaque point de la surface par rapport aux données. Il suffit d'enregistrer dans une carte de même paramétrisation que la grille testée, chaque point dont le voisinage local (un voxel) de la surface contient un point de l'objet, pour produire une  $s_f$ -carte de contact. La carte proprement dite (appelons-la  $s_f$ -carte) est donc binaire, le 1 correspondant à un point en contact avec les données. On dispose alors de données de contact sur les  $s_f$ -points. On déduit alors une  $s_t$ -carte de contact pour la subdivision de travail, sachant que si plus de deux  $s_f$ -points au contact ont même  $s_t$ -parent, ce dernier est à son tour considéré en contact. Les  $s_t$ -points qui ne sont pas en contact sont alors raffinés.

A l'issue, certains  $s_t$ -points sont donc raffinés, et ce sont eux qui seront déplacés au cours des contractions convexes, hyperboliques et concaves. Les positions des  $s_t$ -points

non raffinés restent déduites de la subdivision à l'itération  $s_t$  de la structure de contrôle.

### 3.5.3- Raffinement quelconque

Il pourrait être envisageable d'avoir des points mobiles appartenant à des subdivisions différentes, et de les déplacer de concert. Il n'y aurait pas de subdivision de travail particulière, puisque toutes pourraient être concernées à la fois, en raffinant un ou plusieurs "fils" (au sens de l'algorithme de subdivision) d'un point devenant immobile. On se heurte en fait à plusieurs problèmes :

- soient deux points mobiles appartenant à deux subdivisions différentes,  $P_s$  et  $P_{s'}$  avec  $s < s'$  ( $s$  est moins fine que  $s'$ ). Si  $P_{s'}$  appartient aux points de contrôle engendrés par  $P_s$ , alors les mouvements de  $P_s$  peuvent être sans effet à cause de ceux de  $P_{s'}$ , surtout si beaucoup de points de contrôle issus de  $P_s$  sont raffinés.  $P_s$  devient alors incontrôlable.

- l'étape de test doit indiquer, pour chaque subdivision, quels sont les points générant une intersection, ce qui est peu efficace en temps de calcul.

- les opérations de subdivision ayant lieu conjointement au déplacement, il est difficile de définir quand un point est immobile. Une déformation lointaine peut avoir un effet de bord plus ou moins immédiat.

- Le blocage d'un point mobile est sujet à caution, si on raffine par la suite un de ses voisins, dont le déplacement permettrait d'optimiser la position du point stoppé prématurément.

## 4- Contraintes externes

### 4.1- Prise en compte de l'environnement

La clé de l'évolution de la surface réside dans la "perception", ou l'acquisition de renseignements sur son environnement. Une partie importante de l'algorithme d'évolution va donc consister en cette acquisition, permettant de situer la surface par rapport aux obstacles situés dans l'espace de travail.

La surface plongée dans le volume de données suit le format défini dans le chapitre 3, et sera donc évaluée à une certaine subdivision  $s_f$  : ce niveau de subdivision doit être suffisamment élevé pour que le polyèdre associé approxime finement la surface réelle. La subdivision finale adoptée doit être telle que la distance maximale entre la surface et son approximation soit plus petite qu'un pixel. Mais n'oublions pas que subdiviser une grille de contrôle coûte cher en temps de calcul : le nombre de subdivisions d'une évaluation devra être le plus réduit possible.

De plus, on ne peut pas changer impunément de subdivision finale au cours de "la vie" de la surface : aussi choisira-t-on une subdivision "assez forte" (en pratique 5 itérations), en sur-subdivisant localement après convergence du  $s_f$ -polyèdre. On a vu en effet, dans le chapitre 3, comment accroître localement la précision de la surface. La création de cette nouvelle surface interviendra uniquement dans le cas où l'approximation polyédrique finale est trop grossière. Soit  $(f_i)$  l'ensemble des faces du  $s_f$ -polyèdre qui concourent au  $s_f$ -point P. On considère que l'approximation est mauvaise en P quand deux faces  $f_i$  et  $f_j$  ont des normales dont l'angle est "trop ouvert" (en pratique 30 degrés). Ceci n'est cependant vérifié que pour une surface de degré supérieur à 2, et si le point P n'est pas multiple, ce qui induit une dégénérescence de la continuité géométrique en ce point.

La contrainte externe proprement dite va être prise en compte au moyen d'une procédure de test, dont le "cahier des charges" est le suivant :

- indiquer s'il y a intersection avec les objets de la scène ou non : cette fonctionnalité va permettre à l'étape de traitement de réagir, soit en continuant sa contraction, soit en revenant sur sa position antérieure.

- s'il y a intersection, fournir une carte de la surface indiquant les zones d'intersection. Compte tenu du fait que l'on s'occupe du déplacement des points de contrôle par subdivision croissante, il s'agit de fournir des indications d'intersection pour les points de la subdivision de travail, alors que le test d'intersection proprement dit de la surface s'effectue sur la subdivision finale.

La réalisation d'une telle procédure est alors aisée. La surface étant évaluée à la subdivision  $s_f$ , on dispose d'un ensemble de grilles de points correspondant à la subdivision  $s_f$  des grilles de contrôle de chaque produit tensoriel constituant la (ou les) surface(s). Il suffit de tester leur intersection avec les points de contour représentés par la structure arborescente. Dans le cas d'un graphe de contrôle, c'est le  $s_f$ -graphe que l'on va tester. Le nombre de  $s_f$ -points peut cependant ne pas être suffisant pour "couvrir" toute la surface. On effectue alors trois tests en cascade :

- tester uniquement les  $s_f$ -points.
- tester un "fil de fer", construit à partir des  $s_f$ -points.
- tester un ensemble de "patches bilinéaires", construit à partir des  $s_f$ -points.

Le premier test permet de vérifier rapidement si la surface intersecte l'objet, ce qui est le cas le plus fréquent.

La première réponse au cahier des charges est alors évidente : s'il existe une cellule de la bitmap dont les coordonnées sont celles d'un point généré par la procédure de test utilisée, il y a intersection.

La production d'une carte d'intersection repose sur le même principe que la recherche des

raffinements. On produit une  $s_f$ -carte contenant chaque point qui a engendré un contact. La carte proprement dite est binaire, le 1 correspondant à un point intersectant les données. On recherche cependant une carte ( $s_t$ -carte) associée à la subdivision de travail. Pour cela, on enregistrera tous les  $s_t$ -parents (au sens de la subdivision) des  $s_f$ -points enregistrés dans la  $s_f$ -carte.

#### 4.2- Contrôle du déplacement

On sait désormais quels sont les critères qui permettent de prédire le déplacement (effet des contraintes internes) d'un  $s_t$ -point. Voyons ici comment réagir, si le déplacement de ce point crée une intersection (effet de la contrainte externe). Avant le déplacement, la surface est à l'extérieur de l'objet, et cette condition constitue l'invariant de la méthode. Soit un  $s_t$ -point, et soit un vecteur déplacement, prédit d'une manière quelconque. Deux cas peuvent se présenter :

- le  $s_t$ -point est responsable d'une intersection avec les données. Sa position à l'itération précédente de la boucle d'évolution ayant été conservée, on est en mesure de le ramener (nous dirons le reculer) à cette position, de façon à ressortir de l'objet traité. En effet, la surface doit rester à l'extérieur de l'objet. Mais on va chercher, par le biais d'une progression dichotomique, à optimiser sa position : ce point reculera donc seulement de la moitié de son vecteur déplacement.

- le  $s_t$ -point a trouvé une position satisfaisante (au sens de l'intersection). Trois sous-cas sont alors à traiter :

- \* le  $s_t$ -point venait de reculer. On l'avance de la moitié de son vecteur recul précédent, pour essayer de "coller" un peu plus aux données.

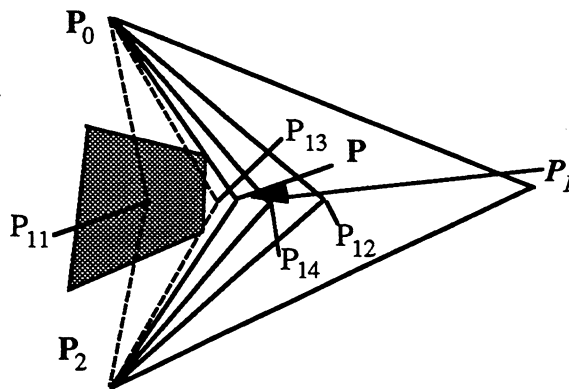


Figure 20 : progression dichotomique d'un point du polyèdre

- \* le  $s_t$ -point venait d'avancer. Si ce n'est pas son premier déplacement dans cette direction (c'est à dire que son premier déplacement a produit une intersection),

on continue d'avancer mais de distance moitié. A l'opposé, si c'est son premier déplacement, c'est qu'il vient de rejoindre la position prédite, et le déplacement est terminé (la prédiction ne crée pas d'intersection).

\* le  $s_t$ -point n'a pas encore avancé : c'est son premier déplacement, et il a donc pour direction et amplitude le vecteur déplacement prédit initialement.

La figure 20 montre un exemple de cette progression : on considère un polyèdre ( $P_0, P_1, P_2$ ), et on déplace  $P_1$ . Pour une meilleure compréhension, on montre sur cette figure le polyèdre seulement, et on prend comme invariant de boucle "le polygone est à l'extérieur de l'objet". Les  $P_{1j}$  montrent les positions intermédiaires du point  $P_1$ .

Dans tous les cas, si le déplacement à réaliser est plus petit que la précision de définition des données, on considère être arrivé à une position convenable.

Un  $s_t$ -point est considéré comme immobile, quand il n'a pu se déplacer à une itération donnée. De plus, si à l'itération suivante, tous ses voisins sont aussi immobiles, on ne cherchera pas à le déplacer. En effet, si son voisinage est figé, la courbure locale de la surface l'est aussi, et il ne pourra pas trouver une meilleure position.

## 5- Conclusion

On a vu comment déformer une surface spline, partant d'une forme pseudo-sphérique, vers la surface externe d'un objet (entièrement défini). C'est un algorithme simulant une "nappe élastique", qui détermine successivement l'enveloppe convexe, l'enveloppe localement convexe, et enfin la surface externe de l'objet proprement dite. De par l'utilisation de la courbure locale de la surface, on est donc en mesure de fournir une contrainte interne suffisamment forte pour s'affranchir du calcul explicite de la distance surface/objet.

Au passage, on montre comment détecter une séparation en deux sous-surfaces (si l'objet est double), et un trou (si l'objet est torique, par exemple). Par ce biais, on obtient une classification de la scène ainsi traitée, puisqu'au gré des transformations topologiques, on peut déterminer le nombre d'objets de la scène, et la topologie de chacun d'entre eux.

Le principe de l'algorithme est simple, puisqu'il consiste à prédire le mouvement des points contrôlant la surface déformable, puis à corriger ce mouvement en fonction des intersections surface/objet.

## **CHAPITRE 5**

### **Application à la segmentation**

## 1- Introduction

Nous avons utilisé dans le chapitre précédent des données de type bitmap volumique pour tester les déformations. Maintenant, on cherche à appliquer l'outil ainsi développé sur des données réelles. Après un regard sur les différents types d'acquisition d'images tridimensionnelles, on va montrer que l'on peut rejoindre le cas précédent en construisant une bitmap volumique à partir des données brutes.

Mais, comme la détection peut se révéler incomplète ou fautive, on adaptera la méthode de déformation pour qu'elle puisse corriger la bitmap en fonction de critères portant sur les niveaux de gris.

Enfin, on verra rapidement dans quelle mesure l'outil se prête à l'intégration de connaissance *a priori* en montrant sur un test de faisabilité comment on peut déformer un modèle d'un objet sur un objet de même classe.

## 2- Données à segmenter

### 2.1- Types de données 3D

La structure des données réelles tridimensionnelles utilisées dépend étroitement du mode d'acquisition. Il faut dans un premier temps différencier deux types d'acquisition tridimensionnelle : l'acquisition surfacique et l'acquisition volumique. Nous n'utiliserons pas de données provenant d'une acquisition surfacique, qui revient à prendre des vues externes d'un objet sous différents angles.

Pour l'acquisition volumique, on distingue plusieurs manières d'obtenir des données. Elles utilisent des variations d'énergie d'un front d'onde ayant la propriété de traverser les matériaux et d'être sensible à une propriété physique de ces matériaux. Par suite, elles sont en mesure de renseigner sur le volume traversé, en étudiant le signal reçu en fonction de cette propriété physique. Par exemple, en  $\gamma$ -scintigraphie, on intègre les valeurs des pixels émettant le long d'une droite donnée.

Les différentes possibilités de balayage induisent trois grands modes d'acquisition :

Le mode projectif (figure 1), donne des "vues intégrantes" sur différents angles. C'est le cas des projections radiographiques obtenues en tournant autour d'un objet. Chaque pixel des images intègre toutes les densités réparties entre sa position et la source émettrice. Dans notre cas, nous n'utilisons cependant pas encore ce type d'images, l'outil développé "travaillant" sur un véritable volume. En effet, la structure de notre outil est adaptée à un

univers volumique, et non aux projections de cet univers.

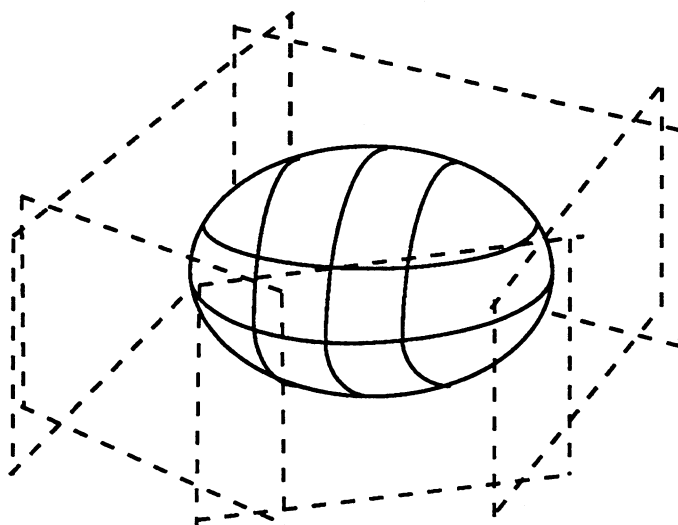


Figure 1 : balayage en mode projectif

Le mode à coupes parallèles (figure 2) réalise une acquisition volumique d'une "tranche" de la scène, où chaque pixel de chaque coupe contient l'intégration de la caractéristique étudiée dans l'élément de volume correspondant. L'Imagerie par Résonance Magnétique est un bon exemple d'une telle modalité d'imagerie. La reconstruction du volume de données est alors immédiate puisqu'il suffit d'empiler les images des coupes. En fait, selon le matériel utilisé les "tranches" d'une épaisseur donnée peuvent se chevaucher, être juxtaposées ou disjointes.

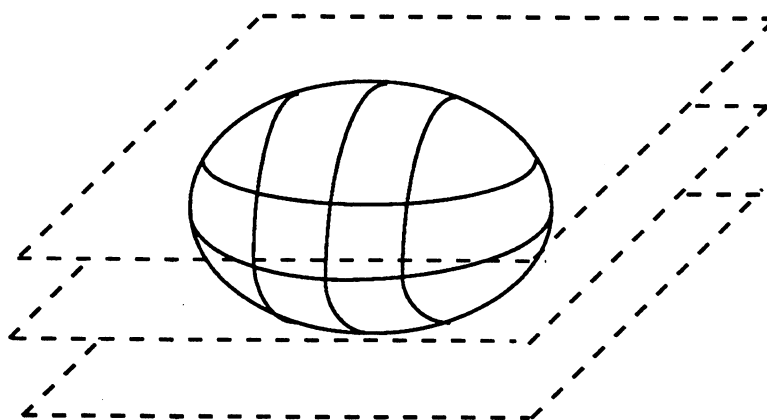


Figure 2 : mode à coupes parrallèles

Le mode à coupes non parallèles (figure 3) fournit des données similaires au modèle parallèle. La reconstruction du volume est ici plus délicate que la simple juxtaposition puisque deux points de deux images différentes peuvent avoir mêmes coordonnées spatiales, mais on peut replacer ces images en position, en faisant une moyenne sur les points confondus. Il faudrait alors interpoler circulairement, mais là encore la nature interpolante des surfaces employées devrait supprimer ce problème. Nous n'avons pas



encore utilisé ce type de données, la définition de l'opérateur de détection de contours (basé sur le gradient) étant plus délicate dans ce cas qu'avec le mode à coupes parallèles. Nous présentons par contre un exemple de reconstruction de données échographiques.

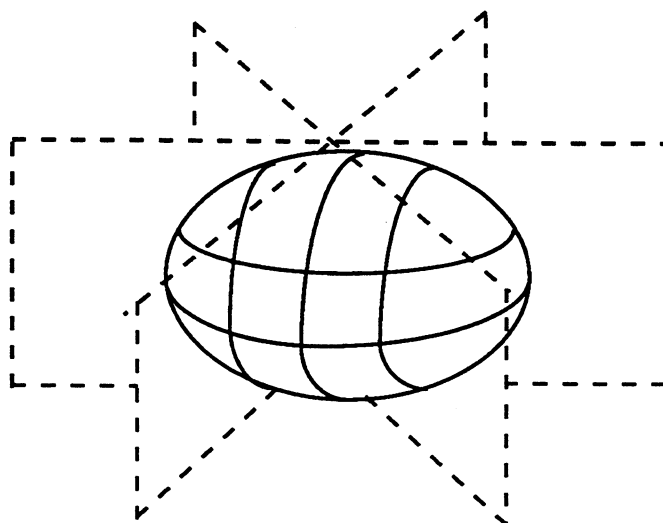


Figure 3 : mode à coupes non parallèles

## 2.2- L'effet de volume partiel

Dans la suite du chapitre, on va contraindre une surface déformable à représenter un objet plongé dans un volume de données en niveaux de gris. Pratiquement, ce volume est construit par l'empilement de coupes successives d'une scène formant des images en niveaux de gris. En fait les données ne sont pas isotropes, mais comme on utilise une surface continue, il suffit de se placer dans un repère orthogonal, avec un facteur réducteur dans l'axe des coupes. Il faudrait tenir compte par ailleurs de l'épaisseur des coupes qui induit un effet de volume partiel (dû à l'intégration des densités sur l'épaisseur de la coupe), mais cet effet est négligé pour la raison suivante :

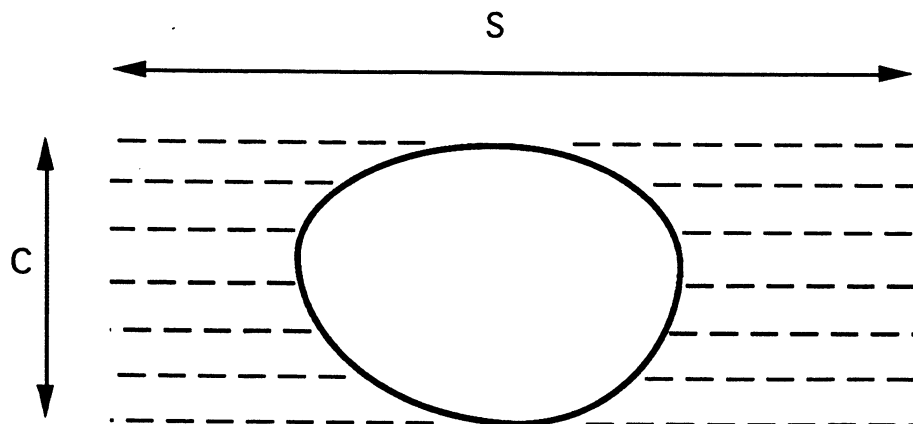
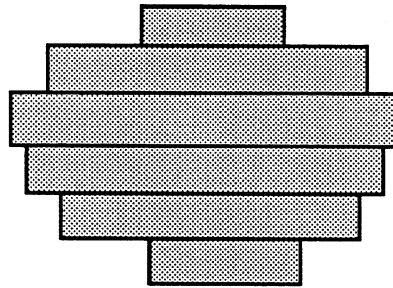


Figure 4 : objet test pour une acquisition parallèle

Examinons l'exemple suivant (figure 4), pris pour simplifier le problème sur une image bidimensionnelle de taille  $S \times C$ .  $C$  (nombre de coupes), vaut ici 6, et  $S$  est choisi très grand devant  $C$ .

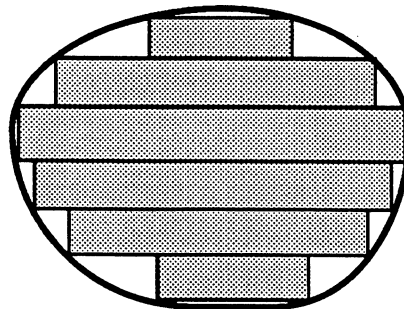
L'objet est un espèce d'oeuf, placé de telle sorte que ses plus grandes tangentes soient alignées dans le sens des coupes.

L'image étant binaire, on considère qu'une cellule de l'appareil d'acquisition colorie le pixel correspondant quand la densité à cette cellule correspond à plus de 0.7 fois (par exemple) la densité de l'oeuf. On obtient alors une image discrétisée de la forme suivante (figure 5):



*Figure 5 : résultat discrétisé de l'acquisition*

Cependant la méthode de mise en contrainte de la surface fournit une enveloppe de tous ces pixels coloriés (figure 6). Compte tenu des caractéristiques géométriques de la bulle, le résultat sera bien meilleur en termes de continuité que l'ensemble des pixels du contour.



*Figure 6 : utilisation d'un contour paramétré sur des données discrètes*

Mais ce résultat reste faux par rapport à l'objet précédent à cause d'une discrétisation excessive de l'acquisition. Dans le cas d'images en niveaux de gris, un pixel contient en fait la somme des densités des matières pondérées par leur volume sur un élément de coupe. Un contour tangent à la direction des coupes produit donc un flou dans les niveaux de gris, qui peut tout aussi bien être interprété comme un réel changement de densité sans rapport avec l'effet de volume partiel.

Aussi il faudrait utiliser un détecteur de contour capable de discriminer ce cas. En fait, la qualité et la résolution des équipements d'acquisition de données allant croissant, la finesse croissante des coupes va éliminer petit à petit l'importance de cette distorsion. Par suite, il n'est pas rentable de développer un détecteur fin de l'effet de volume partiel, la précision du calcul de segmentation dépendant de la précision de l'acquisition. Par ailleurs, décider d'un effet de volume partiel sur un pixel demande à connaître déjà la configuration géométrique des densités de la scène (principalement la tangence d'un contour à un plan de coupe), ce qui n'est pas faisable sans un minimum d'informations sur la scène acquise.

### **3- Détection d'indices de contours**

#### **3.1- Méthode**

Les images sur lesquelles nous allons essayer l'outil de surfaces déformables étant livrées brutes, il est nécessaire d'y appliquer un traitement permettant d'extraire des données à ordonner. Ces données sont évidemment en relation étroite avec les contours pré-supposés des objets à détecter. Aussi va-t-on dans un premier temps utiliser un détecteur de contour classique (gradient) qui va sur-segmenter les images. L'utilisation des lignes de crête du gradient (sur un petit voisinage) va permettre d'affiner la position des contours, et enfin un débruitage partiel sera réalisé en éliminant les lignes de crête spatialement trop réduites. Ceci nous donnera des indices de contour.

On s'appuie sur deux hypothèses, vérifiées de manière assez fréquente dans le cas d'images médicales :

- le phénomène de bruit est souvent de type impulsionnel, qui se traduit par des fausses détections d'éléments très petits.

- les "vrais" contours sont assez bien marqués, et surtout souvent très lisses (sauf "pathologies"). Un problème fréquent consiste en la détection de contours très épais, en partie à cause de l'effet de volume partiel.

#### **3.2- Détection de lignes de crête du gradient**

Dans un premier temps, on effectue un calcul de la norme du gradient sur les données, qui va dégager les variations d'intensité en niveaux de gris. On utilise pour cela un opérateur de Sobel 3D [ZUC], sur un voisinage de 3 pixels d'arête. A cette étape, rien n'empêche cependant l'utilisation d'un autre opérateur [CAN] [MAR]. Les contours apparents sont généralement représentés par les plus fortes variations, et on peut

envisager de déterminer un seuil en deçà duquel on négligera le gradient : les points de "gradient fort" restants seraient donc candidats à être de bons contours. La détermination et l'existence même de ce seuil restent cependant problématiques. On cherche de plus à obtenir des éléments de contour d'épaisseur minimale (1 pixel) , et l'utilisation d'un seuillage ne garantit pas la minimalité de cette épaisseur.

On va donc effectuer une recherche des lignes de crête du gradient (en cherchant les points qui maximisent la norme du gradient dans une direction et dans un voisinage de 3 pixels d'arête) de façon à obtenir des données binaires, qui respectent la condition d'épaisseur ci-dessus. Notons que cette technique ne supprime en rien le bruit résidant dans les données brutes, et qu'elle peut supprimer abusivement des contours réels très proches (pour une distance inférieure ou égale à un pixel).

### **3.3- Construction d'éléments de surface (ou indices)**

On dispose maintenant d'un ensemble de points qui appartiennent à des lignes de crêtes de la fonction gradient des données (dans un certain voisinage). Ceci revient donc à obtenir les points des contours des objets contenus dans les données, mais de manière plus précise que le simple seuillage du gradient. Cependant, on cherche à supprimer une partie du bruit, sans trop perdre d'information pertinente. Pour cela, on part de l'hypothèse que les contours faiblement marqués ne représentent pas l'information principale des données à traiter. Aussi va-t-on supprimer les lignes de crête trop petites, grâce à un seuil de taille. Le bruit devrait donc être ainsi éliminé en grande partie.

Pour cela, on va regrouper les points appartenant aux lignes de crête sous forme d'éléments de surface, et contrôler la taille de chacun d'entre eux. En deçà d'une certaine taille, ils sont éliminés. La méthode de construction de tels éléments de surface est la suivante.

On va chercher à regrouper des points de type ligne de crête du gradient situés sur des éléments de surface. Le choix entre plusieurs configurations de groupement de points est réalisé en conservant celui qui est le plus homogène en termes de niveaux de gris. Ceci s'écrit sous la forme d'un algorithme de suivi de surface élémentaire, exécuté sur l'ensemble des points crête, tant qu'il existe des points non traités :

-pour le point crête courant, choisir les voisins non traités (non marqués) qui forment avec lui une surface élémentaire dans un voisinage de 3 pixels d'arête : on interdit l'apparition d'un volume de 2 pixels d'arête. En cas de possibilités multiples, on choisit les voisins tels que la moyenne en niveau de gris du groupe soit la plus proche de celle du

point courant. La figure 7 donne un exemple sur un carreau 3x3 (en deux dimensions pour être plus clair), d'un cas multiple, et d'un cas ne générant pas un contour assez fin (carré 2x2).

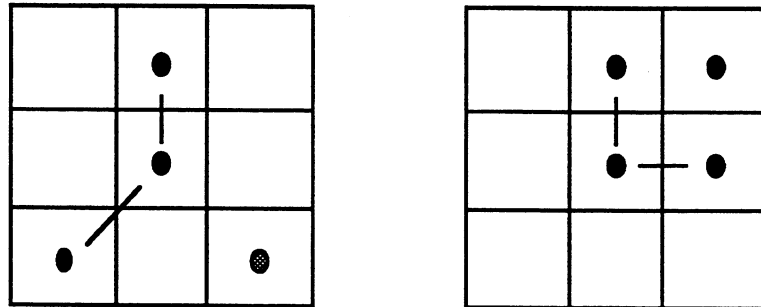


Figure 7 : cas de contour multiple et cas de contour trop épais (en 2D)

-marquer le point crête courant en remplissant une bitmap volumique. La structure de la bitmap est évidemment la même que celle utilisée dans le chapitre 4. L'utilisation de cette bitmap sera d'ailleurs aussi la même.

-relancer récursivement sur tous les voisins sélectionnés, ou arrêter la récursion lorsque le voisinage du point courant est vide ou entièrement marqué.

A la fin de cet algorithme, on contrôle la taille de l'élément obtenu au moyen de deux critères :

-le premier teste les deux plus grandes arêtes du parallélépipède englobant l'élément (mis-à-jour au cours de la récursion). En deçà d'une valeur minimale, on supprime l'élément. En pratique la valeur du seuil vaut 5 pixels (1% de la taille de l'image), mais il apparaît qu'au delà il n'y a pas d'amélioration notable sur les fausses détections.

-le second critère est un critère de remplissage de l'élément. Pour être représentatif, il doit être suffisamment dense, aussi on compte le nombre de points sur l'élément, et on le compare à la moitié du produit des deux plus petites arêtes du parallélépipède englobant. S'il est plus petit, on juge que l'élément de surface n'est pas assez "dense" pour représenter une vraie surface. Ceci permet d'éliminer des suites de pixels "en dentelles", qui n'ont généralement rien à voir avec des contours robustes.

Par rapport aux hypothèses d'un marquage assez fort des contours pertinents, le résultat obtenu contient donc des éléments de surface assimilés à des indices de contours exploitables par la suite. Il est important de noter que le calcul précédent peut être fait "en masse" sur tous les points crête détectés sur le gradient du volume de données, mais qu'il serait plus judicieux de l'effectuer "à la demande", au passage de la surface. C'est cette dernière solution qui est à préconiser pour réduire le temps de calcul. De plus, ce sont les

points de la surface qui vont désigner les points crête initiaux de chaque suivi de surface élémentaire.

On utilise donc ici des lignes de crête de la fonction gradient, mais le passage à zéro du Laplacien des images convient tout autant pour détecter les contours. Dans une variante de la méthode présentée, on utilise le Laplacien nul pour compléter les lignes de crête. Notons le problème de la précision : compte tenu de la discrétisation, les passages à zéro du Laplacien ne sont pas nécessairement au même endroit que les crêtes du gradient.

#### **4- Mise en cohérence de ces contours**

##### **4.1- Mise en place d'une surface déformable**

Disposant d'indices de contours de taille convenable, il s'agit de placer la surface initiale dans leur repère, et d'actionner ensuite le mécanisme de déformation. En fin de déformation, elle aura rejoint un certain nombre d' indices, en les ordonnant explicitement. En principe, si la détection précédente est convenable, on obtiendra les objets voulus.

Sur un plan pratique, deux stratégies de placement sont possibles :

- la première consiste à placer l'enveloppe déformable à l'extérieur de tous les indices de contour détectés, et de laisser évoluer le système, qui se dédoublera éventuellement si plusieurs objets appartiennent à la scène. Plus générale, cette solution est forcément plus lente si on cherche un objet petit par rapport à la scène. Elle est de plus sensible aux mauvaises détections, et ce d'autant plus que les objets à détecter sont nombreux. Peu importe la forme de l'enveloppe initiale, puisqu'on n'a pas d'information sur le lieu et la forme de ce qu'on cherche : pratiquement on utilise un objet de type sphérique.

- la seconde stratégie, plus pratique, consiste à placer la surface autour de l'objet dont on veut détailler les contours. Cette solution demande cependant un interfaçage permettant de choisir la région d'intérêt. Le calcul des indices de contours doit alors se limiter à cette région. Bien que non générale, cette méthode n'est cependant pas à négliger, car l'initialisation peut de plus se faire sur la forme de l'objet à détecter. On reviendra sur cet aspect des choses dans la partie connaissance *a priori*.

Dans un premier temps, on choisit la première méthode, qui demande peu d'interaction, en supposant que la détection des indices de contours ne posera pas de problèmes. Par la suite, il semble incontournable de se référer à la deuxième méthode, qui devrait permettre d'obtenir un résultat quelle que soit la qualité des images : en effet, une initialisation fine

et un contrôle de l'utilisateur tout du long de l'évolution permettra d'éviter des objets collés ou des zones parasitées (le cas extrême reviendrait à guider "à la main" tous les points de contrôle de la surface déformable).

#### **4.2- Evolution de cette surface**

La surface étant initialisée en une certaine position, la méthode de déformation reste équivalente à celle que l'on a vue dans le chapitre précédent. La différence essentielle porte sur l'environnement de la surface. La bitmap que l'on a utilisée était jusqu'à présent figée : on lui permet maintenant de modifier son contenu, pour s'adapter à la position et la cohérence de la surface :

-l'adaptation à la position est la suivante : la bitmap est initialement vide, et va être remplie à chaque arrêt de la surface active sur un point crête. Cela revient à détecter les indices de contours "utiles". En effet, le principe étant que la surface "palpe" les contours le long de sa surface à chaque itération, il n'est pas utile de déterminer tous les indices de contours présents dans le volume de données. Sur un plan pratique, cette détermination se fera donc dans la phase de prise en compte des contraintes externes : après avoir testé l'intersection entre la position prédite de la surface et les indices de contour connus (c'est-à-dire déjà détectés), on exécute une recherche d'indice (à partir des crêtes, selon le principe du paragraphe 3.3) pour tous les points de la surface situés sur des points crête du gradient.

-l'adaptation à la cohérence suppose que les contours d'un objet sont de préférence sur les mêmes niveaux de gris, ou du moins qu'il n'y a pas de brusques différences de teintes. Ce choix s'appuie sur le fait que les objets à rechercher sont généralement des tissus composés de cellules de type homogène, qui devraient avoir une réponse équivalente à l'acquisition des images. Malheureusement, il apparaît cependant que cette réponse varie en fonction de leur position spatiale (en IRM par exemple), et que rien n'empêche d'avoir un bruit de même nature que les tissus étudiés. Malgré tout on utilisera ce critère, et on détaillera dans les deux paragraphes suivants comment éliminer des indices de contours détectés au cours de l'avance de la surface, ou rajouter des indices de contours n'apparaissant pas sur les crêtes du gradient.

#### **4.3- Elimination d'indices de contours**

L'objectif de cette procédure est d'éliminer des indices indésirables. On a vu que le critère est relatif à une certaine homogénéité en niveaux de gris de la surface. Notons qu'il est

possible de lui substituer un autre critère et de conserver le principe de la suppression d'indices.

Un point important est celui de savoir quand effectuer ce test. Alors que la détection d'indices à partir des crêtes du gradient se fait à chaque déplacement de la surface, le test de suppression ne peut être mis en oeuvre que de manière plus épisodique pour une simple raison d'efficacité. En effet, la détection d'un indice ne se fait qu'une fois, alors qu'un test sur les niveaux de gris des pixels où se situe la surface est différent pour chaque position de cette dernière. On se limitera donc à effectuer le test en fin de contraction (convexe, hyperbolique, concave), la surface étant le plus proche possible des indices.

A ces moments là, on calcule la moyenne des niveaux de gris des pixels appartenant à la fois à la surface et à un indice. Comme la surface n'est jamais "sur" un indice, ceci revient à prendre les points de la bitmap à une distance de la surface plus petite ou égale à 1 pixel.

Muni de la moyenne globale et de son écart-type, on réalise un deuxième passage sur ces points, et on repère les points s'écartant de plus de 2 écarts-types (appelons-les "points extrêmes"). Pour chacun de ces points, on parcourt récursivement l'indice auquel ils appartiennent (par connexité), et si la moyenne de cet indice est aussi différente de la moyenne globale que l'est le point extrême correspondant, on juge cet indice indésirable. L'élimination de l'indice de la bitmap se fait par le même parcours récursif, et on supprime au passage les références correspondantes sur les points crêtes.

Pour la surface, il est alors possible d'éliminer des raffinements éventuels en rapport avec cet indice, dans le but de retrouver de la rigidité là où elle a été adaptée. On reprend alors l'algorithme de déformation au niveau de subdivision minimum.

#### **4.4- Création d'indices de contours**

Les détecteurs de bas niveau peuvent non seulement produire de faux indices de contours, mais aussi ne pas exhiber certains contours, trop "flous" (d'où apparition éventuelle de "trous parasites"). Ceci est fort gênant, la surface pouvant pénétrer alors à l'intérieur de l'objet.

Pour ne pas trop pénaliser en temps de calcul la contraction de la surface, on n'effectuera une recherche de ce genre d'artefacts qu'avant de procéder à la création de concavités dans la surface. Précisons tout de suite que cette technique ne prétend pas éliminer tous les problèmes de ce genre. En effet, si une concavité réelle de l'objet est mal fermée, il



n'y a aucun moyen de savoir si cette concavité amorce un trou dans la surface, ou si elle a manqué d'indices pour la définir.

La méthode est toujours basée sur des critères d'homogénéité sur les niveaux de gris des contours. A chaque création de concavité (appelons C le point de contrôle qui crée la concavité), il faut calculer les moyennes locales en niveau de gris sur la surface avant et après création de la concavité. On dispose à ce moment sur la grille de contrôle du point C, entouré de points "non concaves". En parcourant ce voisinage, on peut alors calculer ces moyennes locales, pour chacune des deux positions successives de C (avant et après création de concavité). Il reste à comparer ces moyennes à la moyenne globale, obtenue précédemment : si la première moyenne et non la seconde ne diffère pas de la moyenne globale par plus d'un écart-type, on juge que la concavité est erronée.

On termine alors le processus en créant des indices de contours dans la bitmap au niveau de la portion de surface contrôlée par C que l'on vient d'étudier, et en ramenant C à sa position initiale.

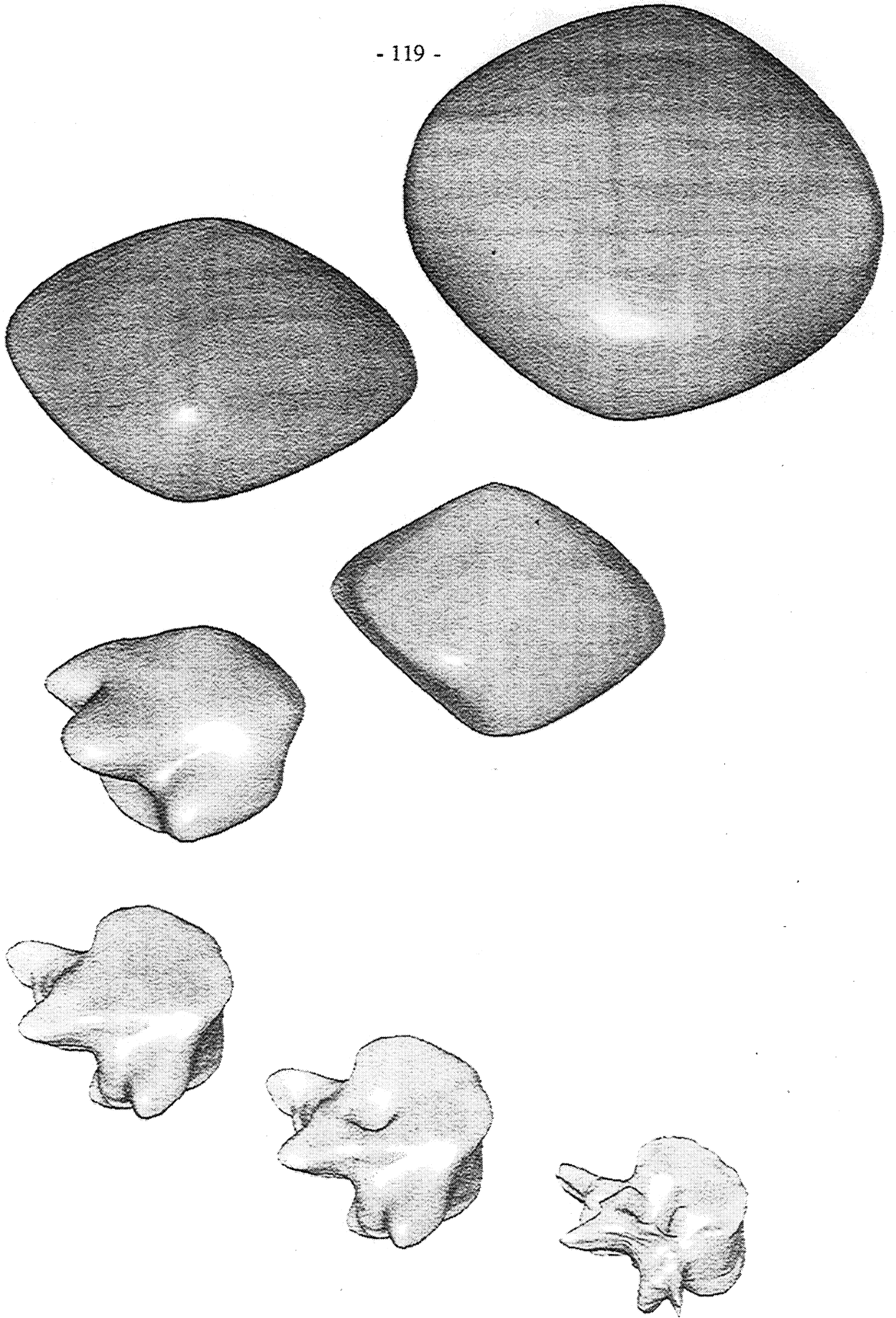
#### **4.5- Résultats**

Dans ce paragraphe, nous montrons une application de la méthode sur des objets médicaux.

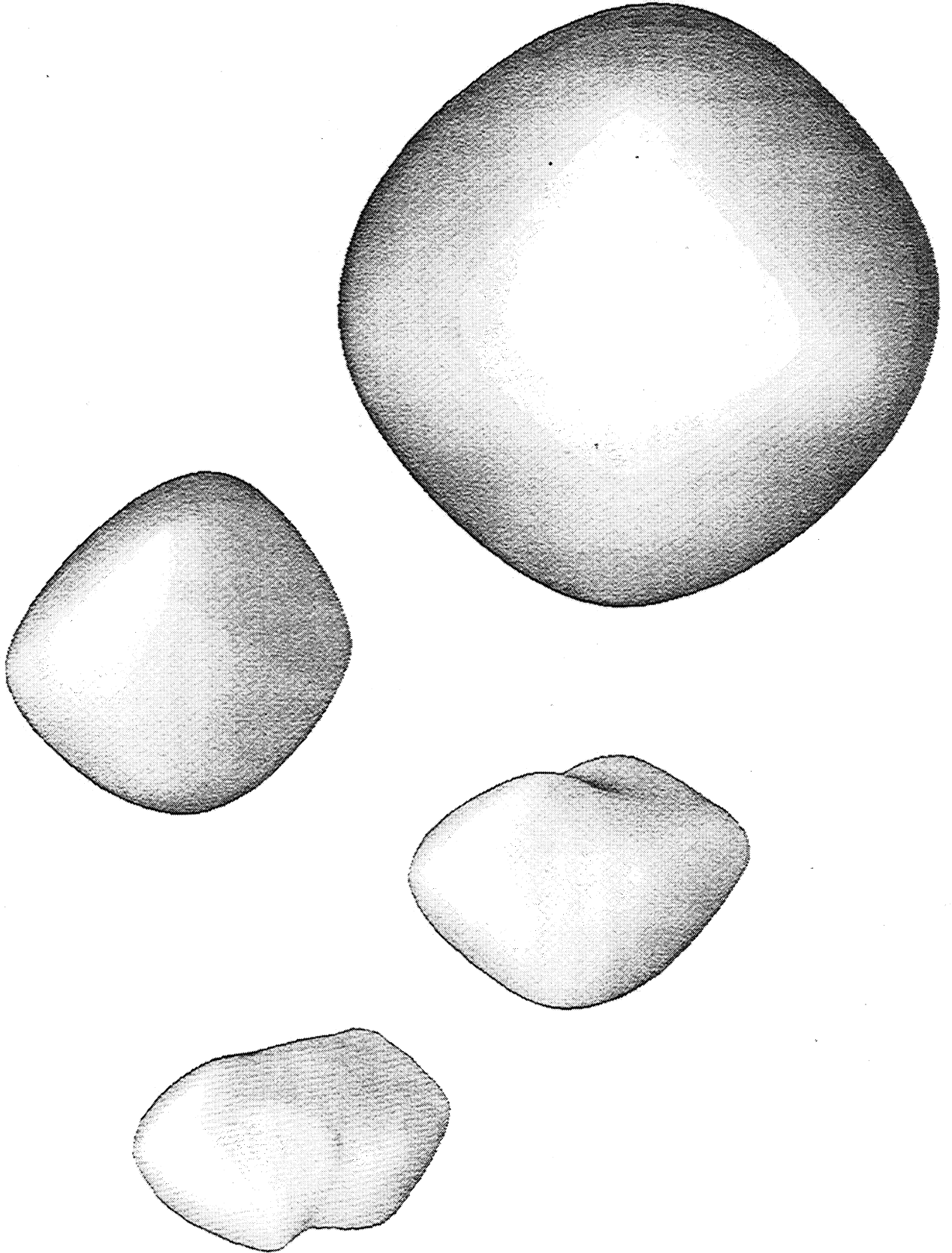
La figure 8 montre ainsi le résultat de l'évolution d'une surface fermée bicubique sur un examen scanner d'une vertèbre lombaire. Le jeu de données utilisées comporte une cinquantaine de coupes axiales. Remarquons que l'extrême enchevêtrement des vertèbres entre elles nous a amenés à sélectionner manuellement des contours communs à deux vertèbres. En effet, l'utilisateur connaît la structure d'une colonne vertébrale, tandis que le programme n'en est pas encore là...

L'exemple précédent utilise des coupes parallèles, aussi voyons une utilisation de coupes quelconques, à savoir l'examen échographique d'un rein [MOE]. Le problème est ici multiple. D'une part, l'imagerie échographique est telle que chaque coupe a été segmentée manuellement. D'autre part, les coupes obtenues ne définissent pas forcément l'objet de manière complète. A cet effet, la figure 10 montre les repositionnements dans l'espace des coupes de deux examens de rein.

Compte tenu de ces données, et de la forme spécifique au rein, on va exécuter l'algorithme en limitant volontairement le nombre de coefficients splines d'une part, et en ne retenant que les étapes de contraction convexe et hyperbolique. Ceci a pour effet d'empêcher la surface de "se glisser" entre les coupes. Sur la figure 9, on voit le résultat d'un tel procédé, exécuté sur l'examen le plus fourni en données.



*Figure 8: Contraction de la surface fermée initiale sur un examen de vertèbre*



*Figure 9: Contraction de la surface fermée initiale sur un examen de rein*

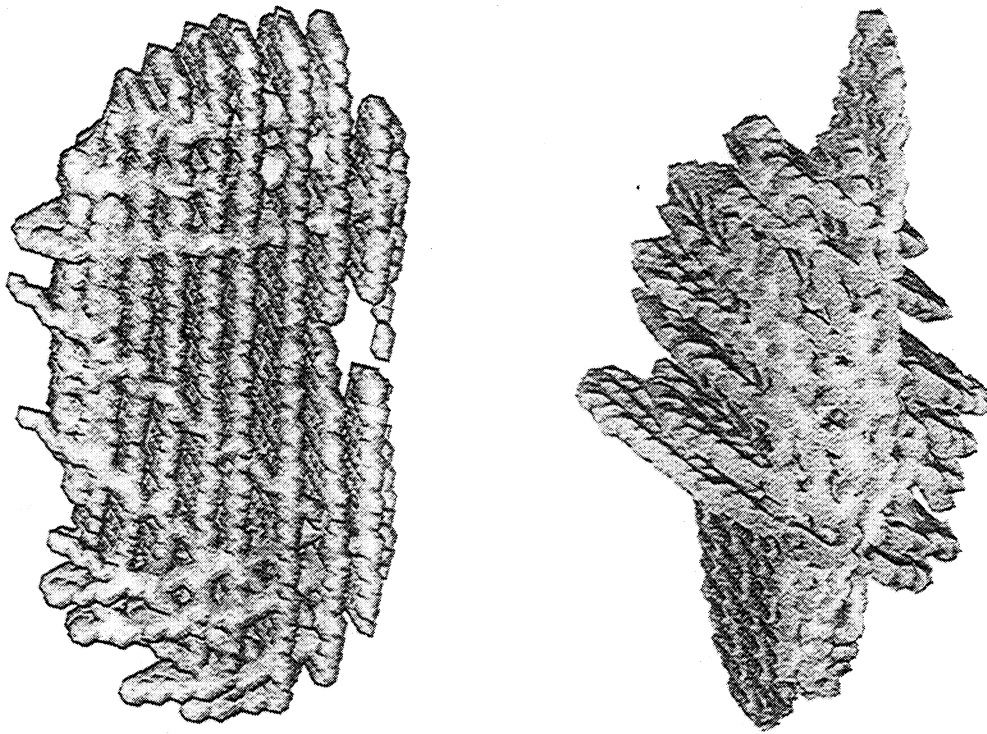


Figure 10: Repositionnement spatial de coupes échographiques de reins

## 5- Introduction de connaissance à priori

### 5.1- Principe

Comment utiliser la connaissance *a priori* dans l'étape de segmentation? Supposons que l'on dispose d'un modèle de l'objet à segmenter, défini au moyen des surfaces présentées dans ce travail. Ce format de données assure un nombre de points de contrôle réduit et une rigidité adaptée. Remarquons qu'il est aisé d'obtenir un tel modèle, puisqu'il suffit de segmenter un volume de données non bruité, contenant seulement l'objet à modéliser.

L'idée de base consiste à initialiser la méthode de segmentation actuelle avec un tel modèle, ce qui rejoint la seconde stratégie d'initialisation vue dans le paragraphe 4.1. Ensuite, il s'agit de déformer sur un principe analogue le modèle pour l'adapter aux données.

On s'aperçoit que la partie "initialisation" proprement dite est assez ardue à réaliser, au sens où il faut déterminer un certain nombre de paramètres :

- la position du modèle, que l'on peut repérer par son centre de gravité, soit trois paramètres.

- l'orientation, référencée par trois angles.
- un (ou plusieurs) coefficient(s) de dilatation, le principal étant l'échelle du modèle.

De tous ces paramètres, le plus difficile à régler semble être celui du facteur d'échelle. En effet, une position ou une orientation imparfaite sera "rattrapée" par l'algorithme de déformation. On va donc laisser l'étape d'initialisation, et surtout l'échelle du modèle à la charge de l'utilisateur, dans un premier temps...

Par contre, on expose dans le paragraphe suivant une solution pour déformer un modèle, qui n'a pas la "chance" d'être initialisé entièrement à l'extérieur de l'objet à modéliser.

## 5.2- Adaptation d'une forme prédéfinie

On présente ici l'adaptation d'un modèle à un objet à modéliser. La différence notable par rapport à la méthode générale présentée jusqu'à présent réside en deux points :

- l'initialisation utilise un objet non banal (à l'opposé de la forme pseudo sphérique).
- la surface du modèle n'est pas entièrement externe à l'objet.

Notons que l'exemple présenté ici ne représente qu'une étude de faisabilité.

La surface de l'objet à reconnaître est stockée sous forme bitmap, et donc est parfaitement connue. Le modèle étant initialisé dans une position identique à celle de l'objet, on voudrait réutiliser au mieux l'algorithme de convergence utilisé jusqu'à présent. Or, la boucle de l'algorithme a pour invariant d'être toujours à l'extérieur de l'objet. Aussi, deux solutions viennent à l'esprit :

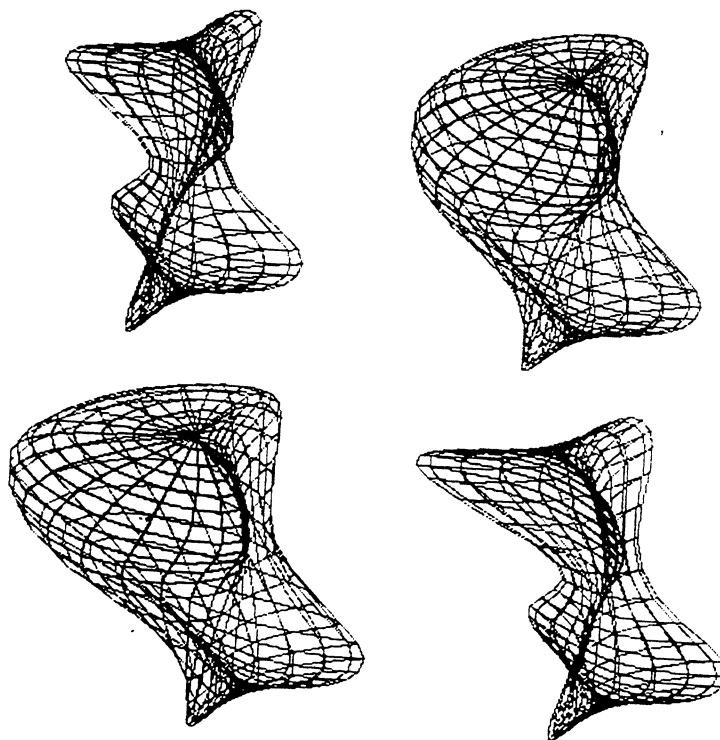
- utiliser une première passe qui "ramène" les zones du modèle à l'extérieur de l'objet. Il suffira ensuite d'exécuter dans une deuxième passe l'algorithme actuel.
- modifier l'algorithme actuel pour que les points internes progressent de manière à augmenter le volume de l'objet, en les empêchant de sortir de l'objet.

On choisit la première méthode ( pour un premier temps du moins) pour une raison de simplicité de développement par rapport à l'existant. En effet, la seconde solution nécessite de savoir exactement si un point est à l'intérieur, ou à l'extérieur de l'objet. Or on n'a à notre disposition que le prédicat "intersecte(surface, données)" (avec de plus les points intersectants).

Une image de la première solution consiste à étirer vers l'extérieur les parties qui intersectent, et de relâcher élastiquement la surface une fois qu'elle est à l'extérieur. A partir du prédicat "intersecte(surface, données)", on peut donc étirer la surface de proche en proche.

Pratiquement, on va supprimer (par subdivision décroissante) les raffinements qui créent une intersection, puis déplacer les points de la subdivision zéro s'il le faut. Par ailleurs, pour conserver la structure du modèle, on peut garder une trace des raffinements disparus : dans l'étape de contraction, on peut forcer l'algorithme à ne raffiner qu'en ces points (ceci n'est pas forcément correct, si le modèle est mal placé par rapport à l'objet).

La méthode développée fonctionne sur le même principe d'une boucle en trois étapes. L'étape d'acquisition reste la même, mais on modifie les étapes de prédiction et de réponse de la manière suivante : supposons que la surface soit raffinée à la subdivision  $s$ . L'analyse de la carte de contact produite par l'étape d'acquisition montre les  $s$ -points qui sont dans le voisinage de la surface. Tous les  $s$ -points raffinés y apparaissant sont éliminés. La surface s'étant modifiée, on répète cette technique tant qu'il y a des  $s$ -points sur la carte de contact. On continue alors avec la subdivision  $s-1$ , jusqu'à la subdivision 0. S'il y a encore intersection, on déplace cette fois-ci les 0-points de manière radiale vers l'extérieur.



*Figure 11 : une transformation élastique d'un modèle vers un objet*

Dans la figure 11, on présente un modèle d'objet à 6 branches, l'objet à modéliser ayant une branche plus longue que le modèle : ce dernier intersecte donc les données. Partant de la forme "modèle" (première forme), certains 1-points sont éliminés (en terme de raffinements). La surface intersectant encore l'objet (deuxième forme), un 0-point

s'écarte radialement du barycentre de l'objet. La surface étant maintenant entièrement à l'extérieur (troisième forme), on réutilise l'algorithme de contraction initial, et on permet aux 1-points de se raffiner à nouveau pour obtenir l'objet (dernière forme).

## 6- Conclusion

L'application de l'outil de surfaces déformables peut donc s'appliquer à la segmentation pourvu que les données ne présentent pas un rapport signal sur bruit trop défavorable.

La gestion des irrégularités de détection (suppression ou rajout d'indices de contours) reste cependant peu fiable dans le cas général, puisqu'elle utilise un critère référençant une moyenne en niveau de gris, ce qui n'a jamais été un critère de choix prédominant. Un critère sur la forme générale de la surface au cours de la déformation serait en fait le bienvenu, et c'est dans cette direction que l'on peut espérer introduire de la connaissance *a priori*.

L'exemple de déformation de modèle précédent n'est cependant pas la panacée, puisqu'il hérite d'une forme initiale mais a peu de contraintes de forme dans la suite de sa déformation. D'ailleurs il reste à définir dans quelle mesure la contrainte de forme à respecter n'empêche pas une complète adaptation.

## **CHAPITRE 6**

### **Conclusion**



## 1- Introduction

Après avoir vu comment construire une surface déformable, et comment lui faire épouser la surface d'un objet, nous avons essayé d'utiliser l'outil ainsi formé comme un opérateur de segmentation de niveau intermédiaire.

C'est donc le moment de conclure d'une part sur le travail réalisé (6.2, 6.3, 6.4) mais surtout de voir quelles sont les perspectives d'applications et de recherches (6.5, 6.6).

## 2- Représentation de surfaces

Le choix de la représentation surfacique des objets à détecter nous ayant amenés à utiliser une représentation sous forme de produits tensoriels de B-splines, plusieurs résultats ont été obtenus :

- on dispose de deux formes de représentation d'une surface fermée : l'une correspond à un cylindre à pôles, défini par une grille de contrôle régulière, munie de conditions aux bords. L'autre utilise un graphe de contrôle, reliant continûment plusieurs grilles.

- la souplesse d'une telle représentation a pu être prouvée grâce à sa véritable adaptation aux formes les plus variées, l'algorithme de calcul utilisé pouvant de plus mélanger de façon transparente des carreaux de degrés différents (par duplication de points de contrôle aux bords).

- la représentation adaptée aux déformations a de plus la possibilité de changer de topologie et ceci représente un des points importants du travail ainsi réalisé. Il y a une perte de continuité quand un tel changement intervient sur une grille régulière, mais c'est de toutes façons un état instable qui peut se corriger par la suite. Notons que c'est la forme à base de graphe de contrôle qui est la mieux adaptée aux changements de topologie, surtout s'ils sont nombreux.

- au delà de l'utilisation de ces surfaces déformables dans le cadre de la détection de contours, qui a pour but d'apporter un outil supplémentaire, on peut noter que la représentation choisie est tout à fait valide pour être utilisée dans le cadre de la CAO, et être utilisée par des logiciels existants (Euclide-MatraDatavision, AVS).

- surtout, elle convient tout à fait pour les applications médicales qui supposent des objets lisses et déformables au cours du temps. De par sa nature, il est facile de calculer les normales à la surface, les courbures, et de repérer les accidents de continuité.

### 3- Déformation de surfaces

La technique de déformation de surfaces développée dans ce travail modélise donc une surface réagissant comme un ballon suivant qu'on le gonfle ou le dégonfle. Partant d'une position externe à la scène à modéliser, on le laisse se dégonfler (par sa propre élasticité) pour atteindre l'enveloppe convexe de la scène, et après une étape intermédiaire l'amenant à obtenir l'enveloppe localement convexe de cette scène, on continue le "dégonflage" dans les concavités qui restent à découvrir (en "aspirant" le volume intérieur).

Cette méthode apporte un élément essentiel, qui est la prise en compte de propriétés géométriques de la scène. En effet on considère la scène comme totalement inconnue, et c'est la démarche progressive utilisée qui va permettre de connaître bon nombre d'informations :

- l'enveloppe convexe donne la forme et la position globale de la scène dans l'espace.

- l'enveloppe localement convexe donne le nombre et la localisation des objets dans la scène, ce type d'enveloppe mettant en valeur les zones convexes des objets.

- enfin la dernière étape fournit la description surfacique exacte des objets, notamment leur topologie (en donnant le nombre de trous par objets).

Ceci n'est cependant vrai que lorsque les contours des objets décrivant la scène sont bien connus.

### 4- Segmentation

Dans le cadre du problème de la segmentation, nous avons été amenés à trouver une méthode permettant de relier des indices de contours, n'ayant dans le meilleur des cas qu'une propriété de connexité. De manière plus générale, on peut voir cette méthode comme un moyen de modéliser l'interpolation naturelle qu'effectue l'oeil sur une suite d'éléments discrets. En fait, ce problème a déjà été abordé dans la classe de méthodes dites des contours actifs, ou modèles déformables. On a proposé ici une variante, fonctionnant sur une modélisation de type spline, selon des déformations de type élastique.

La technique des modèles déformables s'inscrit obligatoirement dans le processus de segmentation, à cause de la cohérence qu'elle apporte aux données traitées avec des opérateurs classiques. La méthode proposée ici obtient par suite des résultats prometteurs sur l'avenir de l'interprétation des contours, le principal apport étant un réel travail en 3D, incluant la résolution des problèmes de changement de topologie. Cependant, il faut être

conscient que cet outil n'est qu'un maillon d'une chaîne beaucoup plus vaste qu'il reste à développer : en effet les résultats vérifient la validité du concept de mise en cohérence d'éléments "sûrs" par une surface interpolante. Mais ceci ne résoud en rien les problèmes de bruit et de connaissance *a priori*, incontournables pour une image quelconque.

La position de ce travail se place donc parmi les opérateurs de niveau intermédiaire, apportant une mise en cohérence d'un certain nombre de caractéristiques géométriques de la fonction densité associée à l'image. Le problème général de la segmentation n'est bien entendu pas résolu, puisqu'il reste à développer d'autres outils de mise en cohérence de même niveau (prise en compte de symétries et/ou périodicités), et surtout à commencer l'analyse des problèmes dits de haut niveau, faisant intervenir la connaissance humaine à proprement parler.

L'apport le plus important de la méthode reste cependant la possibilité d'appréhender un objet complexe, que ce soit au niveau de sa topologie, ou que ce soit au niveau de sa multiplicité (en termes de composantes connexes).

## **5- Applications**

### **5.1- Sculpture virtuelle**

Dans un premier temps, il est possible d'utiliser uniquement les étages de représentation et de déformation de cette méthode, dans un simple but de modélisation surfacique. Ceci forme en effet un outil de maillage à partir de données 3D, ou encore un outil de reconstruction 3D à partir de coupes, en utilisant l'aspect interpolant des fonctions spline utilisées. Il donne de plus des informations quant à la continuité, la normale et la courbure en tout point de la surface.

Cependant on peut n'utiliser que l'étage de représentation dans le cadre d'une interface de "sculpture virtuelle", permettant de travailler avec des surfaces souples de degré et de topologie quelconques (figure 1).

L'intérêt principal consiste en la possibilité de déformer un objet en un autre. Une application médicale pourrait être de prévoir l'allure d'une structure après une intervention, ou de trouver le geste médical pour passer d'une forme à une autre. Il faudra bien sûr modéliser les caractéristiques physiques de cette structure, mais l'étage de représentation des surfaces est déjà fait.

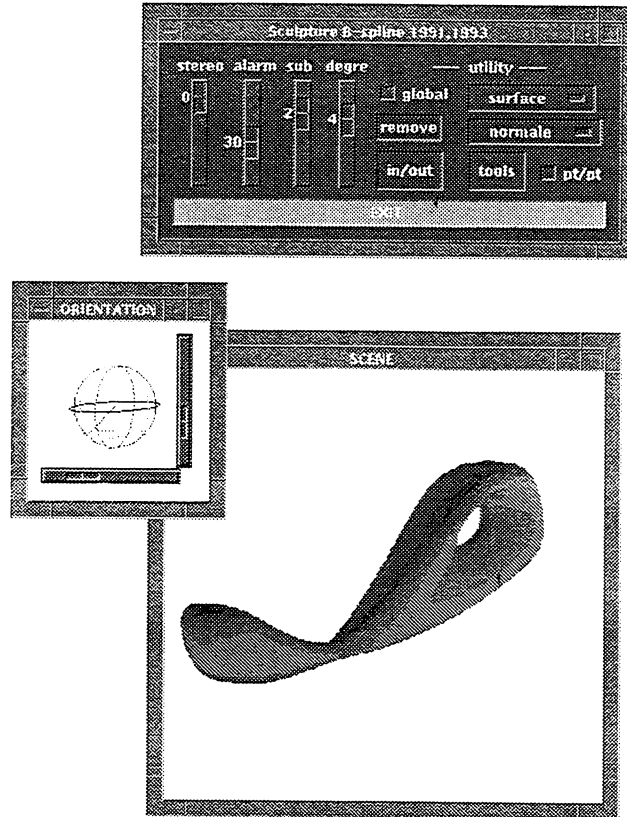


Figure 1: Interface de sculpture virtuelle

## 5.2- Segmentation interactive

Compte tenu des restrictions apportées sur la méthode en tant que technique générale de segmentation, une alternative "facile" consiste à l'intégrer dans une interface utilisateur destinée à apporter les corrections nécessaires.

En effet, résoudre le problème général de la segmentation n'a d'intérêt que lorsque les données à traiter sont de grande taille, ou de nature très répétitive. Dans le cas bidimensionnel on peut segmenter "à la main" quelques images. Cependant, le cas tridimensionnel va devenir de plus en plus courant, grâce à l'apparition de moyens d'acquisition isotropes. La segmentation manuelle d'une "image tridimensionnelle" sera alors fastidieuse : en effet, il faudra traiter, par exemple, l'équivalent d'un lot de 512 images 512x512.

On propose donc un outil dérivé de la méthode présentée ici, qui va segmenter automatiquement une telle image, mais sous contrôle interactif. Lors d'un blocage de la surface, l'utilisateur peut relancer l'évolution de tel ou tel point de contrôle, en supprimant les obstacles parasites. Le travail nécessaire varie donc avec la qualité de l'image. Le logiciel présente donc :

- trois piles d'images petit format correspondant au trois plans de coupe orthogonaux, permettant de choisir une coupe de travail. L'intersection de la surface avec ces coupes y est inscrite.

- une image de travail grand format sur laquelle on peut cliquer des points de l'intersection surface/données, et donc "négocier" le déblocage de la surface. Cette vue permet d'affiner les petites erreurs de segmentation, moins criantes et en principe plus facile à gérer automatiquement.

- une vue tridimensionnelle de la surface permettant de contrôler d'un coup d'oeil la forme globale de l'objet, et d'agir en conséquence : on choisit une zone de la surface, et on déplace un point de contrôle associé (dont la place dans la hiérarchie de contrôle correspond à la taille de la zone à déformer) à volonté. C'est cette manipulation tridimensionnelle qui est la plus importante, puisqu'elle donne un moyen agréable de fournir une connaissance *a priori* sur les données à segmenter.

Ce logiciel est en cours de développement, et illustre bien les problèmes associés au déplacement simultané de plusieurs points de contrôle, surtout dans les premières étapes.

## **6- Perspectives de recherche**

### **6.1- Segmentation**

#### **6.1.1- Exploitation d'une périodicité probable**

Dans le cadre de la segmentation, ce travail pourrait être poursuivi sur l'axe des détecteurs de niveau intermédiaire, qui ne prennent pas en compte des informations de connaissance vis-à-vis de l'objet, mais des mécanismes tels que l'interpolation entre des données.

L'interpolation est plutôt traitée par des mécanismes de mise en cohérence globale tels que la méthode présentée ici, mais il reste toute la partie d'extrapolation à explorer. L'idée consiste à projeter dans les zones de données mal définies une forme probable de l'objet. Cette forme probable est principalement en liaison avec une notion de symétrie ou de périodicité probable de l'objet à détecter. C'est bien sûr faux dans le cas général; mais si cette ressemblance intrinsèque de l'objet existe et qu'une partie seulement de l'objet est bruitée, on pourrait "trier" la zone bruitée et exhiber le contour complet.

L' idée, grossièrement, serait de faire évoluer la surface, et à la fin de l'étape de contraction hyperbolique, évaluer la forme générale de l'objet. De cette forme on peut tirer des conclusions en termes de périodicité et symétrie à partir des éléments "sûrs" de la surface.

Les parties ainsi extrapolées seront alors plongées dans les données que l'on pourra d'ailleurs retraiter en utilisant des opérateurs de taille compatible avec l'information

recherchée, adaptant donc la détection de bas niveau (dans la limite du rapport signal sur bruit bien entendu).

### **6.1.2- Introduction de connaissance *a priori***

Une autre direction d'extension de ce travail consiste à développer un opérateur de niveau supérieur gérant la connaissance nécessaire pour retrouver un objet dans une scène bruitée.

L'information à connaître sera en fait regroupée dans une base de connaissance sur l'objet à détecter, et une manière de stocker cette connaissance peut être d'obtenir un certain nombre d'instances typiques de l'objet à traiter (éventuellement associés à un coefficient de probabilité). L'acquisition de ces objets sera bien sûr réalisée en utilisant la méthode sur des données strictement non bruitées. Un exemple dans le domaine médical est celui du squelette : on peut acquérir des modèles d'os à partir de l'examen scanner d'un mannequin, donnant des images parfaitement "propres".

Le moteur d'inférence d'un tel système cherchera alors à mettre en correspondance (modulo des paramètres d'homothétie) la base de connaissance et la surface courante. Quelques travaux ont été faits sur la manière de déformer une forme en une autre en modifiant l'algorithme d'évolution (5.5). Mais les principaux problèmes résident dans les paramètres d'initialisation, si on veut initialiser la surface à partir d'un objet particulier.

L'introduction de connaissance *a priori* semble cependant la voie la plus prometteuse pour affiner l'étape de segmentation automatique, mais une interface de contrôle interactive sera certainement toujours nécessaire, pour rattraper le système dans des cas trop pathologiques.

### **6.2- Parallélisation**

Dans les étapes de représentation, déformation et segmentation, on sent qu'une bonne part des calculs sont parallèles, voire massivement parallèles.

Une première classe de calculs est bien entendu la détection d'indices de contours : elle fait intervenir une étape de filtrage, puis un regroupement de certains points sous forme d'éléments de surface. Ces opérations sont en effet très locales, et en nombre très important.

La deuxième classe de calcul regroupe toutes les opérations de test. On parcourt la surface

sur la subdivision finale, ce qui impose un grand nombre d'opérations identiques, que ce soit dans le test des coordonnées de chaque point par rapport aux données, ou que ce soit dans le remplissage de la structure d'auto-test.

Une troisième classe de calculs correspond à l'évaluation des points de cette surface. Compte tenu de la nature de l'évaluation par subdivisions de plus en plus fines, et de l'ajout de raffinements, une parallélisation massive n'est intéressante que pour de fortes subdivisions.

Enfin la dernière classe de calculs contient toutes les opérations de déplacement des points. Sachant que l'on déplace un seul point de chaque support à la fois, le travail à des subdivisions faibles est presque séquentiel, et il faut attendre des subdivisions très élevées pour que le nombre de points déplacés simultanément impose un traitement massivement parallèle.

Aussi la plupart des calculs gagneraient à être effectués de manière massivement parallèle, notamment ceux des trois premières classes, qui prennent un temps considérable, et ne représentent pas la partie décisionnelle de la méthode. Cependant, la dernière classe, bien que réclamant un traitement parallèle, est à la limite d'une solution massive, les objets "usuels" ne nécessitant pas des raffinements en nombre trop élevé. On se réfèrera à [PRO], pour le portage sur Maspar (machine massivement parallèle à 8192 processeurs) d'un algorithme de contours actifs de type polyédrique.

**REFERENCES  
BIBLIOGRAPHIQUES**



- [AMI] A.A. Amini, T.E. Weymouth, R.C. Jain, "Using dynamic programming for solving variational problems in vision", IEEE PAMI, vol 12, no 9, septembre 1990.
- [AYA] N. Ayache, J.D. Boissonat, "Building highly structured volume representations in 3D medical images", Proc. of the 3rd Symposium Computer Assisted Radiology, pp 765-772, Berlin, juin 1989.
- [AZI] N.M. Aziz, R. Bata, "Bezier surface/surface intersection", Computer Graphics, pp 50-58, janvier 1990.
- [BAI] E. Bainville, "Reconstruction d'images tridimensionnelles à partir de silhouettes", DEA d'informatique, Université Joseph Fourier, 1992.
- [BAK] H.H. Baker, "Building surfaces of evolution : the weaving wall", Computer Vision, vol 3, pp 51-71, 1989.
- [BAS] M. Basseville, "Détection de contours : méthodes et études comparatives", Ann. Télécommunic., vol 34, no 11-12, 1979.
- [BAR] B.A. Barsky, "A description and evaluation of various 3D models", IEEE CG&A, pp 38-52, janvier 1984.
- [BER] M.O. Berger, "Les contours actifs : modélisation, comportement et convergence", thèse de l'Institut Polytechnique de Lorraine, février 1991.
- [BES] Besl, R.C. Jain, "Segmentation through variable order surface fitting", IEEE PAMI, vol 10, no 2, 1988.
- [BOH] W. Böhm, G. Farin, J. Kahmann, "A survey of curve and surface methods in CAGD", CAGD 1, pp 1-60, 1984.
- [BOI] J.D. Boissonat, "Shape reconstruction from planar cross-sections", CGVIP 44, no 1, octobre 1988.
- [BOO] C. de Boor, "A practical guide to splines", Springer Verlag, Berlin, 1978.
- [BUR] D.J. Burr, "Elastic matching of line drawings", IEEE PAMI, vol 6, pp 708-713, 1981.
- [CAN] J. Canny, "A computational approach to edge detection", IEEE PAMI, vol 8, pp 679-698, novembre 1986.

- [CAT] F. Catté, T. Coll, "Détection d'arêtes par diffusion non linéaire", Conf. On Math. applied to Biology and Medicine, pp 388-389, janvier 1991.
- [CAP] J.P. Cappelletti, A. Rosenfeld, "Three-dimensional boundary following", CGVIP 48, pp 80-92, 1989.
- [CHA] G. Champleboux, "Utilisation de fonctions splines pour la mise au point d'un capteur tridimensionnel sans contact", thèse de l'Université Joseph Fourier de Grenoble, juillet 1991.
- [CIN] P. Cinquin, "Application des fonctions spline au traitement d'images numériques", Thèse d'Etat, Université Joseph Fourier de Grenoble, 1987.
- [COH] I. Cohen, "Modèles déformables 2D et 3D : application à la segmentation d'images médicales", thèse de l'Université Paris IX - Dauphine, 1992.
- [DEL] H. Delinguette, M. Hebert, K. Ikeuchi, "Shape representation and image segmentation using deformable surfaces", IEEE proc of CVPR, pp 467-472, Hawaï, juin 1991.
- [DES] V. Dessenne, "Un outil pour la fusion massivement parallèle d'images médicales multimodales : la carte de distances euclidienne 3D", DEA d'informatique, Université Joseph Fourier, 1993.
- [DU] W.H. Du, "Etude sur la représentation de surfaces complexes : application à la reconstruction de surfaces échantillonnées", thèse de l'ENST, octobre 1988.
- [DUM] G. Dumont, B. Arnaldi, G. Hegron, "Mechanics of solids for computer animation", proc of PIXIM, pp 293-307, 1989.
- [EDE] H. Edelsbrunner, E. Mücke, "Three-dimensional alpha shapes", SIGGRAPH, août 1992.
- [FAR] G. Farin, "From conics to NURBS : a tutorial and survey", Computer graphics, vol 12, pp 78-86, septembre 1992.
- [FLE] Y. Fletcher, D.F. McAllister, "Automatic tension ajustement for interpolatory splines", IEEE CG&A, janvier 1990.
- [FOR] D.R. Forshey, R.H. Bartels, "Hierarchical B-spline refinement", Computer

graphics, vol 22, no 4, août 1988.

- [FRA] O. François, "Systèmes de spins synchronisés", Thèse de l'Université Joseph Fourier, février 1992.
- [FRN] D. Francillard, "Contours actifs", Rapport de Magistère de l'Université Joseph Fourier, 1988.
- [FUC] H. Fuchs et al, "Interactive, richly cued shaded display of multiple 3D objects in medical images", proc of SPIE, vol 914, Medical Imaging II, pp 842-849, 1988.
- [FUC] H. Fuchs, M. Kedem, S.P. Uselton, "Optimal surface reconstruction from planar contours", Commun. Ass. Comput. Mach, vol 20, pp 693-702, octobre 1977.
- [GOR] D. Gordon, J. Udupa, "Fast surface tracking in 3D binary images", CVGIP 45, pp 196-214, 1989.
- [GUE] A. Gueziec, "Reconnaissance automatique de surfaces et courbes gauches. Application à l'analyse d'images volumiques", thèse de l'Université Paris XI, Février 1993.
- [HAR] R.M. Haralick, S.G. Shapiro, "Survey : image segmentation techniques", Computer Vision and Image Processing, vol 29, pp 100-132, 1985.
- [HOH1] K.H. Höhne, R. Bernstein, "Shading 3D images from CT using gray-level gradients", IEEE Medical Imaging, vol 5, no 1, pp 45-47, mars 1986.
- [HOH2] K.H. Höhne, M. Bomans, A. Pommert, M. Riemer, U. Tiede, "3D segmentation and display of tomographic imagery", Proc. of Int. Conf. on Pat. Recognition, pp 1271-1276, Rome, novembre 1988.
- [JEO] H. Jeong, C.I. Kim, "Adaptive determination of filter scales for edge detection", IEEE PAMI, vol 14, no 5, pp 579-585, mai 1992.
- [KAR] B. Kartikyan, A. Sarkav, "A unified approach for image segmentation using exact statistics", IEEE CVGIP, vol 48, no 2, 1989.
- [KAS] M. Kass, A. Witkin, D. Terzopoulos, "Snakes: active contour models", Proc. of Int. Conf. on Computer Vision, pp 259-268, Londres, 1987.

- [KEP] E. Keppel, "Approximating complex surface by triangulation of contour lines", IBM Journal Res. Develop., no 19, pp 2-11, janvier 1975.
- [LAU] P.J. Laurent, "Approximation et optimisation", Hermann, Paris, 1972.
- [LEC] Y. Leclerc, "Constructing simple stable descriptions for image partitioning", Computer Vision, vol 3, pp 73-102, 1989.
- [LEI1] F. Leitner, P. Cinquin, "Complex topology 3D objects segmentation", proc of SPIE, Advances in Intelligent Robotics Systems, vol 1609, Boston, novembre 1991.
- [LEI2] F. Leitner, I. Marque, P. Cinquin, "Dynamic segmentation : finding the edge with snake-splines", Curves and Surfaces, Academic Press, pp 279-284, 1990.
- [LIN] W.C. Lin, C.C. Liang, C.T. Chen, "Dynamic elastic interpolation for 3D image reconstruction from serial cross sections", IEEE Medical Imaging, vol 7, no 3, pp 225-232, septembre 1988.
- [LIU] H.K. Liu, "Two and three dimensional boundary detection", Computer Graphics and Image Processing, vol 6, pp 123-134, 1977.
- [LOO] C. Loop, T. DeRose, "Generalized B-spline surfaces of arbitrary topology", ACM, pp 347-356, 1990.
- [LOR] W. Lorensen, H.E. Cline, "Marching cubes : a high resolution 3D surface construction algorithm", Computer Graphics, vol 21, pp 163-169, 1987.
- [MAQ] I. Marque, P. Cinquin, "Boundary detection for 3D images by construction of a differential system", 7th congress AFCET-INRIA, pp 907-922, Paris, 1989.
- [MAR] D. Marr, E. Hildreth, "Theory of edge detection", Proc R. Soc. London, B 207, pp 187-217, 1980.
- [MEH] A. Le Mehaute, P. Sablonniere, "Courbes et surfaces Bézier / B-splines", A.T.P. du C.N.R.S., Rennes, mai 1987.
- [MIC] C.A. Michelli, "Subdivision algorithms for curves and surfaces", Congrès SIGGRAPH, Dallas, 1986.

- [MON1] O. Monga, R. Deriche, "3D edge detection using recursive filtering, application to scanner images", Proc. of IEEE Conf. on Pattern Recognition, San Diego, juin 1989.
- [MON2] O. Monga, R. Deriche, G. Malandrain, J.P. Cocquerez, "Recursive Filtering and edge closing : two primary tools for 3D edge detection", Proc. of 1st European Conf. On Computer Vision, Nice, mars 1990.
- [MOE] X. Moreau-Gaudry, "Fusion d'images rénales échographiques et tomoscintigraphiques", DEA de Génie Biologique et Médical, 1993.
- [MOR] M. Morgenthaler, A. Rosenfeld, "Multidimensional edge detection by hypersurface fitting", IEEE PAMI, vol 3, pp 482-486, juillet 1981.
- [MUR] MURIM, "Multidimensional Reconstruction and Imaging in Medicine: state of the art report", EEC DG XIII, Advanced Informatics in Medicine, janvier 1990.
- [PEN] A. Pentland, S. Sclaroff, "Closed-form solutions for physically based modelling and recognition", IEEE PAMI, vol 13, pp 715-729, 1991.
- [PHI] M.B. Phillips, G. Odell, "An algorithm for locating and displaying the intersection of two arbitrary surfaces", IEEE CG&A, pp 48-58, septembre 1984.
- [PIE] L. Piegl, "On NURBS : a survey", IEEE CG&A, vol 11, no 1, pp 55-71, janvier 1991.
- [PAT] N. Patrikalakis, "Surface-to-surface intersections", Computer Graphics, vol 13, pp 89-95, janvier 1993.
- [PER] P. Perona, J. Malik, "Scale-space and edge detection using anisotropic diffusion", IEEE PAMI, vol 12, no 7, juillet 1987.
- [PRO] E. Promayon, "Segmentation par contours actifs sur machine massivement parallèle", DEA d'informatique, INP Grenoble, 1993.
- [RAY] S.P. Raya, "Low-level segmentation of 3D Magnetic Resonance Brain Images - A rule based system", IEEE Medical Imaging, vol 9, no 3, septembre 1990.

- [RHO] M. Rhodes et al, "An application of computer graphics and networks to anatomic model and prosthesis manufacturing", IEEE CG&A, pp 12-25, 1987.
- [RIC] M.J. Richard, J. Allard, S.K. Ghosh, M. Bougouss, "Three dimensional reconstruction of human limbs from tomographic views", Computers and Biomedical Research 22, pp 26-35, 1989.
- [ROU1] N. Rougon, "Eléments pour la reconnaissance de formes tridimensionnelles déformables. Application à l'imagerie biomédicale", thèse de l'ENST, février 1993.
- [ROU2] N. Rougon, F Prêteux, "Deformable markers : mathematical morphology for active contour models control", proc of SPIE, Image Algebra and Morphological Image Processing II, vol 1568, 1991.
- [SAB] P. Sablonnière, "Spline and Bézier polygons associated with a polynomial spline curve", Computer Aided Design 10, pp 257-261, 1978.
- [SAN] Sabin, Malcolm, "Non-rectangular surface patches suitable for inclusion in a B-spline surface", Proc of Eurographics '83, pp 57-69, 1983.
- [SEQ] J. Sequeira, F. Pinson, "Matching free-form primitives with 3D medical data to represent organs and anatomical structures", NATO ASI series, 3D Imaging in Medicine, Springer Verlag, pp 159-171, 1990.
- [SHI] P.E. Shile, M.P. Chwialkowsky, D. Pfeifer, R.W. Parkey, "Automated identification of the spine in Magnetic Resonance Images : a reference point for automated processing", CAR 89, pp 678-690, 1989.
- [SHU] L.L. Shumaker, "Spline functions : basic theory", John Wiley & sons, New York, 1981.
- [SZE] R. Szeliski, "Fast surface interpolation using hierarchical basis functions", IEEE PAMI, vol 12, no 6, juin 1990.
- [TER] D. Terzopoulos, A. Witkin, M. Kass, "Constraints on deformable models : recovering 3D shape and non rigid motion", Artificial Intell., vol 36, pp 91-123, 1988.
- [UDU] J.K. Udupa, V.G. Ajjanagadde, "Boundary and object labelling in 3D

images", CVGIP 51, pp 355-368, 1990.

- [WAN] Y.F. Wang, J.F. Wang, "Surface reconstruction using deformable models with interior and boundary constraints", IEEE PAMI, vol 14, no 5, pp 572-579, mai 1992.
- [WES] J.S. Weska, " A survey of threshold techniques", IEEE CVGIP, vol 27, pp 259-265, 1978.
- [WIT] A. Witkin, "Scale-space filtering", Proc of IJCAI, pp 1019-1021, 1983.
- [ZHA] J. Zhao, "Extraction d'information tri-dimensionnelle par stereo vision", thèse de l'Université Paul Sabatier de Toulouse, juillet 1989.
- [ZUC] S.W. Zucker, R.M. Hummel, "A three dimensional edge operator", IEEE PAMI, vol 3, pp 324-331, mai 1981.

## **ANNEXES**



## A- Utilisation d'une carte de distance

Nous avons vu dans le chapitre 4 qu'il est possible d'utiliser dans le cadre des contraintes externes une carte de distance, définissant en chaque point de l'espace la distance au plus proche contour. Après l'avoir utilisée dans un premier temps dans le déplacement de la surface active, l'utilisation des courbures locales l'a remplacée, pour une raison d'efficacité en temps de calcul, et de certains problèmes d'ambiguïté (contours à distance égale).

La construction de la carte de distance est réalisée à partir des contours binaires d'un objet, ou de l'ensemble d'indices de contours d'un volume de données en niveau de gris. On peut convoluer ces contours avec une fonction valant 1 en 0 et 0 à l'infini (adaptée à la dimension de l'espace), de type  $1/n$  ou  $1/e^n$ . On consultera [DES] pour la construction de cette carte sur une machine massivement parallèle (Maspar), qui permet d'obtenir un résultat en 15 secondes sur un volume de 64 voxels d'arête.

L'intérêt de cette carte permet de connaître la distance surface-contours, ainsi que la direction de déplacement d'un point de la surface. L'obtention du déplacement d'un point de contrôle P de la surface déformable se fait en utilisant une somme pondérée de toutes les directions des points dépendant de P. Quant au choix du point de contrôle à déplacer, c'est le point contrôlant le point le plus éloigné des contours qui est utilisé. Ceci revient donc à minimiser la distance surface-contours. Compte tenu de la définition de la carte, il peut exister des minima locaux quand des contours sont à égale distance l'un de l'autre. La solution préconisée est alors similaire à ce que l'on a appelé contraction concave dans le chapitre 4.

L'application à la segmentation de la carte de distance est intéressante puisqu'elle permet de relier deux lignes de crête détectées par filtrage en prenant le chemin le moins "cher" (en termes de distance) les liant. Par contre, en cas d'élimination d'indice de contour (voir chapitre 5), il faut recalculer la carte (du moins localement).

En conclusion l'utilisation d'une carte de distance 3D apporte peu de choses au niveau du déplacement de la surface proprement dit, et sera moins efficace pour la détection d'un changement de topologie de la surface : en effet, il va falloir tester à tous moments les accidents de type trou ou étranglement, alors que la démarche par enveloppes successives permet de cantonner ces tests à des intervalles plus réduits. En termes de segmentation, on peut supposer que le temps de recalcul de la carte de distance est faible (bien qu'il faille utiliser une Maspar) et son utilisation reste compétitive.

## B- Algorithme de subdivision de graphe

Nous présentons ici le détail de l'algorithme de subdivision d'un graphe, basé sur l'algorithme de subdivision d'une grille régulière. En effet, si ce dernier est trivial à programmer, la structure de données (facettes, points) d'un graphe rend sa subdivision plus délicate.

Un graphe  $G$  est constitué d'un ensemble de points  $p$ , chacun muni d'une liste de facettes  $liste(p)$  auxquelles il appartient, et de coordonnées  $coord(p)$ .  $G$  dispose aussi d'une liste de facettes  $f$ , chacune munie d'une liste de points  $liste(f)$  la définissant, et d'une référence sur leur centroïde,  $centroïde(f)$ .

L'algorithme se décompose en une étape de duplication des points (Subdivision( $G$ )), puis en autant d'étapes de calculs de centroïdes (Elévation( $G$ )) que l'ordre des B-splines utilisées (On se référera au Chapitre 3 pour les raisons de cette décomposition). En fait, on regroupera la première étape de calcul de centroïde dans l'étape de duplication.

La subdivision au niveau  $s$  d'un graphe  $G$  d'ordre  $p$  se présente alors ainsi :

```

répéter s fois
{
  G = Subdivision(G)
  répéter p-1 fois
  {
    G = Elévation(G)
  }
}

```

La procédure  $G' = \text{Subdivision}(G)$  s'effectue en deux passes, une dupliquant les facettes (Subdiv1), et une dupliquant les points (Subdiv2) :

```

Subdiv1 :   ∀ f ∈ G
{
  centroïde(f) = pc = Créerpoint(G',card (f))
  coord(pc) = barycentre(liste(f))
  ∀ i < card(liste(pc)), facette(pc,i) = Créerfacette(G',4)
}

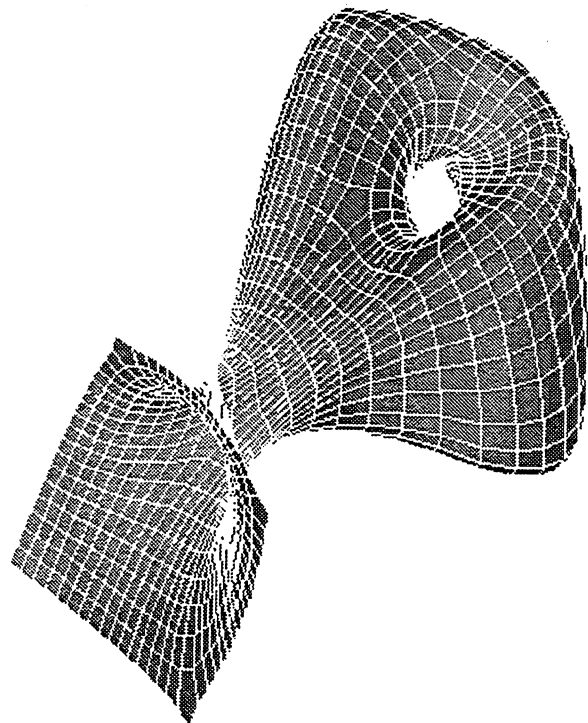
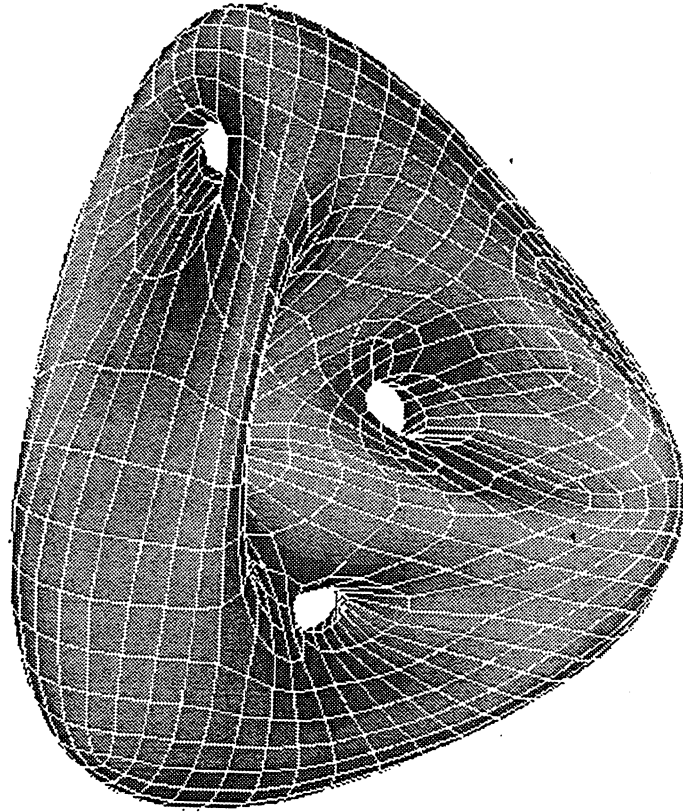
```

où Créerpoint( $G,k$ ) et Créerfacette( $G,k$ ) allouent un point ou une facette munis de  $k$  références sur des facettes ou des points.  $facette(p,k)$  représente la facette numéro  $k$  de  $liste(p)$ .

```

Subdiv2 :   ∀ p ∈ G
{
  p' = Créerpoint(G',card(liste(p)))
  coord(p') = coord(p)
  ∀ fi et fj successives ∈ liste(p)
  {
    i = position de fi dans liste(p)
    j = position de fj dans liste(p)
    facette(p',i) = facette (centroïde(fi),i)
    pm = Créerpoint(G',4)
    coord(pm) = milieu(fi,fj)
    facette(pm,0) = facette(centroïde(fi),i)
  }
}

```



*Subdivision de deux graphes de contrôle*

```
facette(pm,1) = facette(centroïde(fi),i-1)
facette(pm,2) = facette(centroïde(fj),j+1)
facette(pm,3) = facette(centroïde(fj),j)
    }
}
```

La procédure  $G' = \text{Elévation}(G)$  s'effectue en une seule passe:

```
Elévation :  ∀ p ∈ G
             {   f = Créerfacette(G',card(liste(p)))
                 ∀ f ∈ liste(p)
                 {   si pc = centroïde(f) n'existe pas
                     {   centroïde(f) = pc = Créerpoint(G',card(liste(f)))
                         coord(pc) = barycentre(liste(f))
                     }
                     centroïde(f') = pc
                     j = position de f dans liste(p)
                     facette(pc,i) = f
                 }
             }
```

Dans le cas d'un graphe définissant une surface ouverte, les points  $p$  du bord du graphe contiennent dans  $liste(p)$  une facette "vide". Cette facette est ignorée dans Subdiv1, et dupliquée dans Subdiv2 et Elévation. La figure jointe présente deux exemples d'application de l'algorithme de subdivision de graphe, dont un montre un graphe de type plan, raccordé à un graphe de type tore.

### C- Complexité - Implémentation

L'évaluation globale de la complexité de la méthode va suivre le même découpage que le plan de ce travail, en évaluant successivement l'algorithme de subdivision de la surface, la technique de déformation, et enfin les algorithmes de détection de contour (que l'on ne développe pas ici).

La complexité de l'algorithme de subdivision porte sur le nombre de points ( $n$ ) et de facettes (similaire à  $n$ ), le nombre de subdivisions ( $s$ ), et l'ordre de la spline ( $p$ ). L'opération principale est le calcul ( $C$ ) du centroïde d'une facette. On obtient pour cela un temps de calcul pour l'évaluation de la surface de l'ordre de :

$$p.C.(4^s + 4^{s-1} + \dots + 4).O(n)$$

La partie déformation va faire évoluer ces  $n$  points pour adapter la surface aux données. Le temps de calcul principal pour un point consiste à calculer la convexité locale en ce point, calculer le centroïde de ses voisins (adapté selon le type de contraction), soit un temps de calcul  $T$ . On obtient au total, pour une itération, un temps de calcul de l'ordre de  $T.O(n)$ . On rajoute à ce temps, le temps d'évaluation de la surface. On multiplie enfin ce total par le nombre total d'itérations, ce qui varie avec les formes recherchées, soit :

$$nb_{\text{iterations}} \cdot T.p.C.(4^s + 4^{s-1} + \dots + 4).O(n)$$

Le logiciel issu de cette thèse comporte plusieurs volets, puisqu'on peut discerner :

- un module de calcul concernant la subdivision.
- un module de déplacement des points de contrôle.
- un module de détection d'indices de contours de bas niveau.
- un module graphique pouvant servir de coquille aux modules précédents.

Le tout est programmé en C (X11, OSF/Motif pour les parties d'interface) sous UNIX, et développé sur des stations de travail DEC DS3100, DS5000, dernièrement sur station Alpha DEC 3000.

La mémoire utilisée dépend essentiellement d'une part du nombre de points contenus dans la surface, et d'autre part du nombre de coupes de l'examen traité (coupes 512x512 sur 8bits). En charge moyenne, une station de travail de 64 Mo peut ainsi supporter une trentaine de coupes et une subdivision finale de 4 ou 5, soit jusqu'à 6000 points de contrôle.

En termes de temps de calcul, on séparera le temps de chargement/détection d'indices du temps d'évolution de la surface. En effet, on charge les images au fur et à mesure, ce qui

est fortement dépendant du réseau reliant les stations, et donc très variable selon sa charge. Le temps d'évolution de la surface est lui-même dépendant de la forme de l'objet à segmenter. Ainsi (Chapitre 5), le calcul sur le rein s'effectue en 6mn sur une DS5000, et 2.30mn sur station Alpha (64 itérations de calcul sur la dernière subdivision). Le calcul de la vertèbre prend près de 20mn sur DS5000, et environ 10mn sur station Alpha (175 itérations).







## RESUME

Ce travail traite de la segmentation d'images médicales : ce type de problème est difficile à résoudre à cause de la piètre qualité du signal, et de la fréquente nécessité d'avoir une connaissance spécifique. Aussi se restreint-on à des objets de type volumique courants, tels que les os ou des organes massifs. On modélise une surface fermée à l'aide de produits tensoriels de B-splines (utilisés en CAO), et on propose deux formes de représentation : soit un cylindre à pôles, défini par une seule grille de contrôle, soit un graphe de contrôle, reliant continûment plusieurs grilles. Ces représentations sont adaptées aux déformations globales ou locales, et aux changements de topologie. La déformation d'une telle surface est modélisée par le dégonflage d'un ballon : partant d'une forme englobant la scène, on laisse se dégonfler la surface pour atteindre successivement l'enveloppe convexe, puis l'enveloppe localement convexe, et enfin la surface externe de la scène. L'apport principal de cette méthode consiste en la possibilité d'appréhender un objet complexe, par sa topologie, ou par le nombre de ses composantes connexes. La méthode de segmentation utilise alors cette technique sur l'ensemble des indices de contours détectés dans les images. On peut voir cette méthode comme un outil modélisant l'interpolation naturelle de l'oeil. Ce travail se place ainsi parmi les opérateurs de niveau intermédiaire, en apportant la mise en cohérence de certaines propriétés géométriques de la fonction densité associée aux images. Ceci ne résoud cependant pas le problème de connaissance *a priori*, une interface utilisateur pouvant y remédier dans les cas critiques.

**MOTS CLES :** B-splines - surfaces - topologie - contours actifs - segmentation

## ABSTRACT

This work is about segmentation of medical images. This kind of problem is difficult to solve, because the signal is noisy, and we frequently have to use some knowledge about a specific scene. So we take interest in a restrictive class of objects, as bones or massive organs. We build a closed surface using B-splines tensor products (useful in CAD), and we propose two representations of closed surfaces : a cylinder with poles using one control grid, and a control graph where several grids may continuously merge. Both representations are useful for local or global deformations, or for topological transformations. We model the deformation process as a deflating balloon : starting with a shape surrounding the scene, we let it deflate in order to obtain successively the convex hull, the locally convex hull, and finally the external surface. The main interest of this approach consists in its capacity to find information about a complex object, such as topology, or such as the number of connected parts. The segmentation method uses this technique on a set of edge indices, detected in the images. We can see this method as a tool to model the natural interpolation effect of the human eye. This work is therefore a middle-level operator, which gives a global coherence from geometric properties of the images density function. This does not solve the problem of *a priori* knowledge, but a user interface can be used where we need some specific information.

**KEYWORDS :** B-splines - surfaces - topology - active contours - segmentation