THÈSE DE DOCTORAT
DE L'UNIVERSITÉ DE TECHNOLOGIE DE COMPIÈGNE

Département Génie Informatique

Spécialité Infographie

présentée par :

Philippe DECAUDIN

Modélisation par Fusion de Formes 3D
pour la Synthèse d'Images

~

Rendu de Scènes 3D Imitant le Style "Dessin Animé"

---

soutenue le 3 décembre 1996 devant le jury composé de :

M^me^	Marie−Paule GASCUEL	Rapporteur
M.	Jarek ROSSIGNAC	Rapporteur
M^lle^	Sabine COQUILLART	
M^me^	Christine FERNANDEZ	
M.	André GAGALOWICZ	
M.	Francis SCHMITT	
M.	Xiao Wei TU	

Remerciements

Je remercie Madame *Marie-Paule Gascuel* et Monsieur *Jarek Rossignac* d'avoir accepté d'être rapporteurs de cette thèse, et Messieurs *Francis Schmitt* et *Xiao Wei Tu* pour m'avoir fait l'honneur de bien vouloir faire partie de mon jury.

Je remercie également Mlle *Sabine Coquillart*, qui m'a suivi et soutenu pendant ces trois années passées à l'INRIA.

Je remercie Madame *Christine Fernandez*, et Monsieur *André Gagalowicz* qui m'a accueilli dans son laboratoire.

Je tiens à remercier mes collèges thésards, stagiaires et post-docs de Syntim pour leur aide et leur amitié; et tout spécialement Francis Lazarus, Fabrice Neyet, Xavier Provot et Jos Stam.

Je remercie les permanents de Syntim, chercheurs et non-chercheurs ; en particulier Jean-Paul Chièze et Laurence Bourcier pour leur sympathie et leur disponibilité.

Je remercie l'équipe audiovisuelle de l'INRIA et tout particulièrement Arghyro Paouri et Christian Blonz pour leur précieuse collaboration.

Enfin, je remercie mes parents pour leur soutien, et tous mes proches, famille et amis, qui ont su m'encourager.

Table des matières

Table des figures

Introduction

La synthèse d'images est un outil de visualisation utilisé dans de nombreux domaines et de diverses façons. Ainsi, en architecture et en CAO, elle permet, par exemple, de visualiser une scène présentant un bâtiment, une pièce mécanique ou un assemblage de pièces avant que ceux-ci ne soient réalisés. Cette visualisation peut être figée, on obtient alors une vue de la scène telle qu'aurait pu la saisir un appareil photo ; mais elle peut aussi être dynamique en incluant des objets animés, ou en offrant la possibilité de se déplacer autour ou à l'intérieur de la scène, voire d'*interagir* avec elle en autorisant la manipulation des objets présents dans la scène.

Bien qu'ayant de nombreuses caractéristiques communes, les techniques mises en œuvre dépendent beaucoup des applications auxquelles elles sont destinées. Les travaux présentés dans cette thèse ont pour cadre les productions de films d'animation et de mondes virtuels pour les domaines de l'audiovisuel (cinéma, publicité) et de la réalité virtuelle axée "grand public" (mondes interactifs, jeux vidéo). Ils sont scindés en deux parties.

- La première se rapporte aux outils de modélisation d'objets tridimensionnels. Elle constitue l'essentiel du travail fourni pendant la thèse.

- La seconde concerne le rendu dit *non-photoréaliste*.

Les outils de modélisation

Quelque soit l'application visée, pour calculer des images de synthèse il faut disposer d'une représentation en machine des différents objets présents dans la scène que l'on souhaite visualiser. Pour cela, une plateforme logicielle dédiée à la synthèse d'images propose à l'utilisateur une panoplie d'outils de modélisation. Le chapitre 1

fait un tour d'horizon des techniques existantes.

Dans le cadre que l'on s'est fixé, et contrairement aux domaines tels que la CAO, l'utilisateur, qui est généralement un (info)graphiste, ne désir pas contrôler avec précision la forme des objets représentés : seul l'aspect global de l'objet compte. Il va, par exemple, souhaiter que telle partie de l'objet soit arrondie, mais n'a pas besoin que cet arrondi corresponde exactement à un arc de cercle de rayon donné.

Dans ce cadre, une technique de modélisation très utilisée consiste à partir d'un objet simple et à le déformer (un peu comme on déformerait de la pâte à modeler) jusqu'à obtenir la forme voulue. Cette technique est populaire, car elle permet une interaction directe avec l'objet en cours de création et les manipulations nécessaires pour modeler l'objet deviennent très vite intuitive pour l'utilisateur. Des travaux sur la composition d'objets tridimensionnels commencés en stage de DEA et récapitulés dans le chapitre 2 nous ont amenés à proposer d'étendre la palette des outils de déformation mise à disposition du graphiste par deux nouveaux outils :

- L'outil de fusion permet de déformer localement un objet en le *fusionnant* avec une forme 3D simple (sphère, ellipsoïde,...). Deux types d'effets peuvent être obtenus : l'objet est déformé soit en creux, soit en bosse de façon à englober la forme simple. La forme du creux ou de la bosse est alors celle de la forme simple.

- L'outil de flexion permet de tordre un objet plus ou moins localement. C'est un peu comme si l'objet était fait en matière souple et qu'on y insère une armature articulée. Influer sur l'angle de la flexion revient à manipuler l'articulation.

Ces deux outils présentés dans le chapitre 3 peuvent être utilisés pour modeler tout type d'objet[1] en fonction des besoins du graphiste. Mais, il y un cas où l'utilisation de ces outils est particulièrement adaptée, c'est le cas des objets de type *organiques* (corps humains, d'animaux,...).

La modélisation de ce type de forme demeure un problème difficile. C'est essentiellement dû au fait qu'elle requiert un modèle capable de décrire et d'animer les formes "douces" et "lisses" des structures organiques, en particulier au niveau des articulations et de l'effet des muscles. Les techniques de modélisation basées sur la manipulation directe de la surface de l'objet par l'utilisateur ne sont pas adaptées pour cela. En effet, en agissant directement sur la position des sommets pour une

1. Ces outils de déformation sont des déformations de l'espace et peuvent appliquer à tous les objets indépendemment de la représentation utilisée (cf. chapitre 3)

représentation polyédrique ou sur les points de contrôle pour une représentation par morceaux de surface paramétriques, il est difficile d'obtenir ce type de forme et surtout d'animer cette forme de façon satisfaisante (cf. [Bad82]). Nous verrons dans le premier chapitre, certaines approches qui permettent de traiter ce problème.

Nos outils de déformation permettent aussi d'obtenir de telles formes : la forme générale et, en particulier, l'effet des muscles sont obtenus grâce à l'outil de fusion, et les articulations sont obtenues par l'outil de flexion. L'utilisation conjointe de ces deux outils permet de définir une méthode dédiée à la modélisation des formes organiques articulées dont les principaux avantages sont :

- la possibilité d'animer l'objet en agissant sur des paramètres de haut niveau (taille des muscles et angles des articulations), l'objet se déforme en conséquence ;

- le maintien cohérent de la texture appliquée à l'objet quand celui-ci est animé (la texture ne "glisse" pas sur l'objet lorsqu'on agit sur une articulation) ;

- la possibilité de générer un maillage polygonal de la surface de l'objet à différents niveaux de détails (le maillage comporte plus ou moins de facettes).

Le chapitre 4 présente cette méthode et le chapitre 5 décrit l'implémentation que nous en avons faite.

Le rendu non-photoréaliste

Dans le chapitre 7, nous nous intéressons à un autre aspect de la synthèse d'images : le rendu, c'est-à-dire l'algorithme qui va transformer la description d'une scène tridimensionnelle en une image ou une animation correspondant à cette scène vue à travers une caméra virtuelle.

Ces images se veulent, en général, *photoréaliste*, c'est-à-dire imitant le mieux possible ce que verrait une vraie caméra devant une vraie scène. Dans le cadre des applications de type production de films d'animation pour l'audiovisuel, une nouvelle tendance apparaît : le rendu non-photoréaliste. Le but n'est plus d'imiter la réalité, mais d'obtenir des effets plus exotiques (dits *non-photoréalistes*) comme générer des images de la scène telles qu'un peintre aurait pu les dessiner [Mei96].

Nous décrivons un algorithme de rendu qui génère des images imitant le style «dessin animé» traditionnel (ou «bande dessinée») à partir de la description

tridimensionnelle d'une scène fixe ou animée. Pour ce faire, l'algorithme fait appel à des techniques qui permettent :

- de dessiner les contours des objets (profils et arêtes sont dessinés en traits noirs),
- de colorer uniformément les surfaces intérieures à ces contours,
- de faire apparaître sur les objets les ombres propres et les ombres portées dues aux sources de lumière éclairant la scène.

Par rapport aux techniques de dessin animé traditionnelles, l'avantage de notre méthode est d'automatiser la production des images. En fait, on reporte le problème de l'animation en amont : il faut désormais créer une scène avec un modeleur 3D et l'animer. Ce système est certes moins souple que le dessin image par image, car il est difficile de produire des animations aussi "délirantes" que celles vues dans les cartoons avec les outils actuels d'animation en synthèse d'image. Mais, en contrepartie, outre le calcul automatique des images, il devient beaucoup plus facile de conserver la cohérence spatiale des objets dans la scène, et on peut, par exemple, générer facilement des mouvements de rotation d'objets ou de caméra ainsi que des déplacements de sources lumineuses.

En annexe de cette thèse, on trouvera des exemples de réalisations qui nous ont permis de tester les algorithmes que nous avons développés et de se rendre compte concrètement des problèmes qui se posent lors de la production de films ou de mondes interactifs.

Modélisation par fusion de formes

Chapitre 1

La modélisation en synthèse d'images

1.1 L'objectif

Pour produire une image de synthèse d'une scène tridimensionnelle, il est nécessaire d'avoir en machine une représentation des objets qui composent cette scène. Cette représentation doit permettre, d'une part, de caractériser le plus complètement possible un objet, et, d'autre part, elle doit se coder et se manipuler facilement dans un programme informatique. Nous appellerons **modèle** une telle représentation.

Ce modèle va alors pouvoir servir à calculer un rendu de la scène. Mais, il faut aussi prendre en considération un autre aspect : la création des objets. Cette étape est généralement appelée **modélisation**.

A un modèle sont associés divers paramètres qui permettent de décrire un objet de la scène. Cette description peut se faire à différents niveaux :

- Elle peut, par exemple, ne décrire que la géométrie de l'objet : sa forme. Ainsi, une sphère sera caractérisée par son centre et son rayon ou par un ensemble de petites facettes approximant sa surface.

- Le modèle peut aussi inclure une description de la matière de l'objet. Les paramètres photométriques préciseront comment l'objet réfléchit la lumière et

l'aspect texturel lui associera une texture et précisera comment elle est plaquée.

- Pour les objets déformables, le modèle décrira comment la forme de l'objet évolue lorsqu'on agit sur les paramètres contrôlant la déformation. Par exemple, un corps humain se déforme lorsqu'on agit sur l'une articulation constituant son squelette [BPW93].

Les modèles les plus fréquemment rencontrés incluent la description géométrique et la description de la matière, car ces aspects sont nécessaires pour pouvoir calculer un rendu de l'objet (parfois, la matière est réduite à une simple couleur). Même si, bien souvent, on considère que la matière (incluant la texture) est uniquement un paramètre de rendu et se gère alors séparément.

Un modèle associé à un type d'objets déformables est, pour sa part, caractérisé par les paramètres contrôlant la déformation. Ces paramètres doivent être bien choisis : ils ne sont pas redondant, et permettent un contrôle direct et compréhensible de la déformation. De plus, ils ne se contentent pas d'agir sur la forme de l'objet, mais aussi sur sa matière en garantissant, par exemple, que la texture n'évolue pas de façon aberrante lors de la déformation.

Classiquement, l'animation est gérée séparément. Pour un objet déformable, il s'agit de décrire comment varient les paramètres contrôlant sa déformation lors du mouvement. Ainsi, la marche ou la course seront caractérisés par l'évolution au cours du temps des paramètres contrôlant les articulations d'un objet représentant un corps humain ou animal, ce qui permettra d'obtenir les différentes formes géométriques que prend le corps lors du mouvement.

A plus haut niveau, l'objet animé peut aussi réagir à son environnement et contrôler son mouvement en fonction de la tâche que l'on souhaite lui faire faire. Par exemple, faire marcher le corps humain d'un point à un autre en évitant les obstacles, lui faire gravir des marches, etc... . Ici, les techniques utilisées se rapprochent de celles utilisées en l'intelligence artificielle et en automatique/robotique [GH96].

Pour résumer l'approche traditionnelle des modèles en synthèse d'image, elle est cloisonnée en trois sous-domaines : la modélisation géométrique (pour la forme de l'objet), le rendu (pour la matière) et l'animation. Mais, il faut tout de même se souvenir que la forme et la matière d'un objet peuvent évoluer pendant la phase d'animation.

Dans les sections qui suivent, nous allons nous intéresser aux méthodes de modélisation existantes, c'est-à-dire les divers moyens que l'on peut mettre à la disposition d'un utilisateur pour modeler la forme d'un objet. L'objectif est

de donner une vision d'ensemble sans rentrer dans les détails techniques de ces méthodes de modélisation. Nous présenterons aussi succinctement le modèle sous-jacent lorsque ces méthodes nécessitent l'utilisation d'un type de modèle particulier.

1.2 Différentes approches de modélisation

1.2.1 Modélisation déclarative

La modélisation déclarative [DH93] permet à l'utilisateur de déclarer les caractéristiques d'un objet ou d'une scène sans avoir à préciser la manière de les obtenir. Cette déclaration permet de décrire la forme et la structure d'un objet en terme de contraintes et de propriétés topologiques, géométriques, morphologiques, etc... . Cette approche permet une spécification progressive des objets, par raffinements successifs, laissant au logiciel le soin de proposer des solutions, de gérer les informations incomplètes et incohérentes. La déclaration se fait grâce à un langage qui peut être inspiré du langage naturel. On peut, par exemple décrire une pièce en décrivant les objets qu'on peut y trouver (chaises, table, etc...) puis préciser comment ces objets sont situés les uns par rapport aux autres ("table au centre de la pièce", "table posée sur le sol", "chaises sur le sol et autour de la table",...).

1.2.2 Modélisation procédurale

La modélisation procédurale permet de créer un ensemble d'objets par l'intermédiaire de programmes (ou procédures) spécifiques [New75]. L'utilisateur peut agir sur les paramètres d'entrée du programme pour influer sur la forme de l'objet résultant. Ce type de modélisation est très utile pour permettre à l'utilisateur de créer des instances d'objets définis de façon générique. Ainsi, si le programme permet de générer des roues dentées, l'utilisateur pourra créer une instance de roue dentée en spécifiant le nombre de dents, le diamètre de la roue, etc... . La modélisation procédurale est aussi très utile pour créer des objets "exotiques", comme les objets récursifs (ex : les montages fractales), ou les objets dont la surface est générée par un système de règles (ex : les L-systèmes permettent de générer plantes et arbres) [Smi84].

1.2.3 Modélisation par script

La modélisation par script utilise une description complète et exhaustive des objets fournie par l'utilisateur par l'intermédiaire d'un script. Contrairement à la modélisation déclarative, il n'y a ici aucune contrainte a résoudre, tous les paramètres définissant les objets sont décrits explicitement dans le script. C'est le type de modélisation le plus fréquemment utilisé en entrée des logiciels de rendu (ex : les logiciels domaine public de type lancer de rayon *povray* [Tea93] et *rayshade* [Kol91]). Le script peut être tapé directement par l'utilisateur, ou indirectement produit de façon automatique par un programme.

1.2.4 Modélisation automatique

Certaines techniques permettent de modeler un objet automatiquement. Pour cela, un objet réel est placé devant un ou plusieurs capteur(s). L'analyse des données acquises permettent d'extraire les principales caractéristiques de l'objet, et de reconstruire ainsi sa forme.

L'utilisation d'un capteur du type *Cyberware* [SW91] permet d'obtenir un ensemble très dense de points 3D répartis sur la surface de l'objet traité. A partir de ces données, on extrait un ensemble de points pertinents qui vont servir pour recréer la surface de l'objet.

Citons aussi ici la reconstruction automatique du modèle d'un objet à partir d'images de l'objet réel. La difficulté est de retrouver de l'information tridimensionnelle à partir de données 2D. Cette information peut être déduite par corrélation entre différentes vues d'un objet (ex : la vision stéréoscopique) [Mar82].

1.2.5 Modélisation interactive

La modélisation interactive permet à l'utilisateur de contrôler *interactivement* le processus de création des objets tridimensionnels. Il dispose pour cela d'outils et d'une interface graphique. Cette interface lui permet d'agir directement sur les outils et de voir immédiatement leur effet sur l'objet en cours de création [Shn83].

Le principal avantage de ce type de modélisation est de rendre intuitif l'utilisation des outils mis à disposition. En donnant rapidement le résultat de la manipulation des outils, elle incite l'utilisateur découvrir par lui-même le potentiel de chaque outil. Ceci explique que la modélisation interactive soit présente dans la plupart des logiciels de modélisation du commerce, et plus particulièrement dans

les modeleurs dédiés aux applications audiovisuelles. Elle offre à l'utilisateur un raccourci appréciable entre l'étape de conception et l'étape de rendu d'une scène de synthèse 3D en cachant autant que possible les techniques employées pour y parvenir. L'utilisateur n'a plus besoin d'être un informaticien.

1.3 Outils de modélisation

1.3.1 Construction directe de formes 3D

Les méthodes de construction directe de formes 3D permettent à l'utilisateur de partir de rien pour créer son objet.

1.3.1.1 Construction directe de la surface

Une façon de créer un objet est de construire la surface qui sépare l'intérieur de l'extérieur. On utilise alors une représentation surfacique [Far89]. Le modèle utilisé pour représenter la surface va induire les méthodes de construction associées.

- Polyèdres

Le modèle polyédrique est le type de représentation le plus commun. Il utilise un ensemble de sommets/arêtes/polygones pour définir les faces de l'objet. Les objets sont donc des polyèdres [HCV56]. L'utilisateur spécifie interactivement la position de chaque sommet, les liens entre sommets pour former les arêtes sur lesquelles viennent ensuite s'appuyer les facettes polygonales[1].

Dans le cadre de la construction directe de la surface consistant à positionner et à déplacer les sommets du polyèdre, la représentation de formes arrondies n'est pas aisée. On peut, en fait, approximer ces formes arrondies par des micro-facettes, et donc garder une représentation polyédrique, mais il est hors de question de demander à l'utilisateur de spécifier *à la main* chacune de ces facettes. Pour ce faire, on pourra plutôt utiliser les surfaces paramétriques.

1. Notons que cette construction doit produire un polyèdre *valide*, c'est-à-dire, un polyèdre fermé dont chaque arête est partagée par un nombre paire de facettes. Le polyèdre respecte alors la formule d'Euler :

$$V - E + F = 2$$

où V est le nombre de sommets, E le nombre d'arêtes et F le nombre de facettes. Cette formule est valable pour les polyèdres simples (i.e. à topologie sphérique, i.e. sans trou), une généralisation aux polyèdres à trou existe [HCV56].

- Surfaces paramétriques

Une première solution consiste à utiliser des carreaux de surface paramétrique à la place des facettes. Ces surfaces sont contrôlées par un petit ensemble de points que l'utilisateur va pouvoir déplacer pour influer sur l'aspect de la surface, et obtenir ainsi les formes arrondies qu'il souhaite.

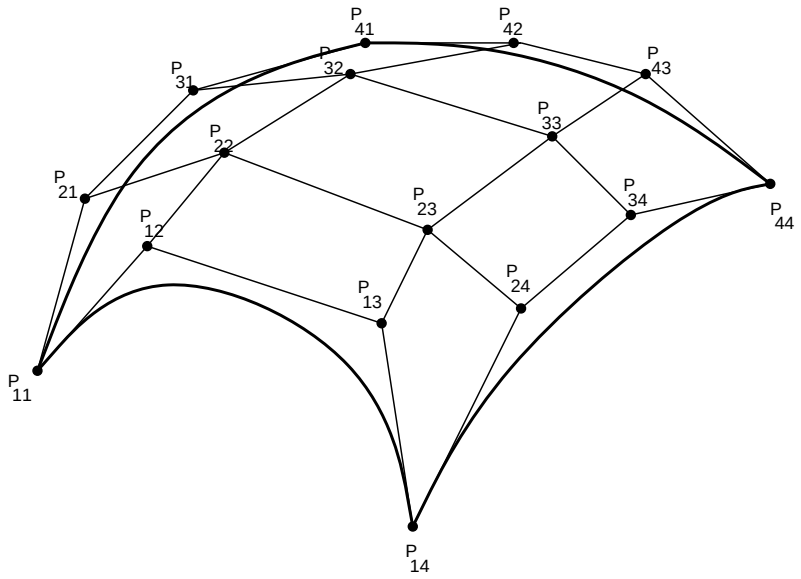

FIG. 1.1 – *Carreaux de Bézier et ses 16 points de contrôle*

Les plus répandues sont les surfaces paramétriques bicubiques telles que les carreaux de Hermite, de Bézier et B-spline [FDFH90, Cob84, Far86]. Ces surfaces sont générées par un polynôme de degré 3 à deux variables, classiquement exprimé de la façon suivante :

$$Q(s,t) = \begin{pmatrix} s^3 & s^2 & s & 1 \end{pmatrix} .M. \begin{pmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \\ P_{41} & P_{42} & P_{43} & P_{44} \end{pmatrix} .M^T. \begin{pmatrix} t^3 \\ t^2 \\ t \\ 1 \end{pmatrix}$$

où M est une matrice 4x4 de réels qui dépend du type de la surface (Hermite, Bézier,...) et les P_{ij} sont les seize points de contrôle de la surface (cf. figure 1.1).

Pour créer un objet complexe, il faut juxtaposer plusieurs carreaux et gérer la continuité entre eux. Si les carreaux sont répartis suivant un maillage régulier quadrangulaire, il est possible d'obtenir une continuité au premier ordre (continuité des normales) en contraignant les points de contrôle.

Les surfaces de type NURBS (Non-Uniform Rational B-Spline) sont obtenues en utilisant des polynômes rationnels dans les surfaces de type B-spline. Elles sont généralement utilisées en CAO car elles permettent d'obtenir une continuité d'ordre 2 entre les carreaux sur un maillage régulier et elles permettent aussi d'obtenir de façon exacte les formes de type conique.

Pour tous ces types de surface, lorsque les carreaux ne sont pas répartis suivant un maillage régulier, il est difficile d'obtenir la continuité à l'ordre un et plus. Il est aussi difficile d'obtenir des objets de topologie arbitraire, car il est alors nécessaire de créer des carreaux ayant un nombre de coté arbitraire [Loo94].

- Subdivisions récursives

Une autre solution, simple et efficace, consiste à partir d'un polyèdre créé par l'utilisateur et à subdiviser chacune de ces facettes en plusieurs, les sommets sont replacés automatiquement de manière à obtenir un polyèdre plus arrondi [CC78, Doo78, Els90]. En répétant plusieurs fois l'opération, on gomme ainsi l'aspect anguleux du polyèdre de départ. Cette technique donne à l'utilisateur l'impression de *lisser* le polyèdre, ce qui la rend assez intuitive.

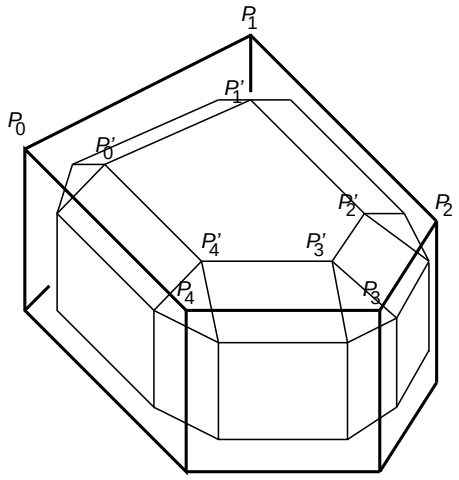

FIG. 1.2 – *Lissage d'un polyèdre : en gras, le polyèdre d'origine ; en léger, le polyèdre après la première étape de lissage.*

L'étape de lissage s'effectue en calculant, pour chaque facette ayant pour sommet $\{P_0, P_1, \cdots, P_{n-1}\}$, un nouvel ensemble de sommets $\{P'_0, P'_1, \cdots, P'_{n-1}\}$ définis par :

$$P'_i = \frac{1}{4}O + \frac{1}{8}P_{i-1} + \frac{1}{2}P_i + \frac{1}{8}P_{i+1}$$

où O est le barycentre des P_i et où les indices sont pris modulo n. Les faces du nouveau polyèdre correspondent à des sommets, arêtes, ou facettes du polyèdre de départ (cf. figure 1.2). Chaque sommet joignant k facettes génère une facette à k cotés, chaque arête génère une facette à 4 côtés, chaque facette à n côtés génère une facette à n côtés.

Dans le cas où les facettes sont organisées en maillage quadrangulaire régulier, on retrouve le schéma de construction récursif des B-splines ; les sommets servent de points de contrôle. Dans le cas général, il s'agit d'un schéma de construction récursif équivalent pour les maillages irréguliers.

1.3.1.2 Les cylindres généralisés

Les méthodes de type cylindre généralisé permettent de construire une surface en faisant glisser un contour (appelé profil) le long d'une courbe. Il s'agit, donc, là aussi, de représentations surfaciques, mais elles sont définies par moins de paramètres que les précédentes ; elles sont donc plus facile à utiliser, mieux adaptées à la modélisation interactive [FDFH90, Klo86, Coq87].

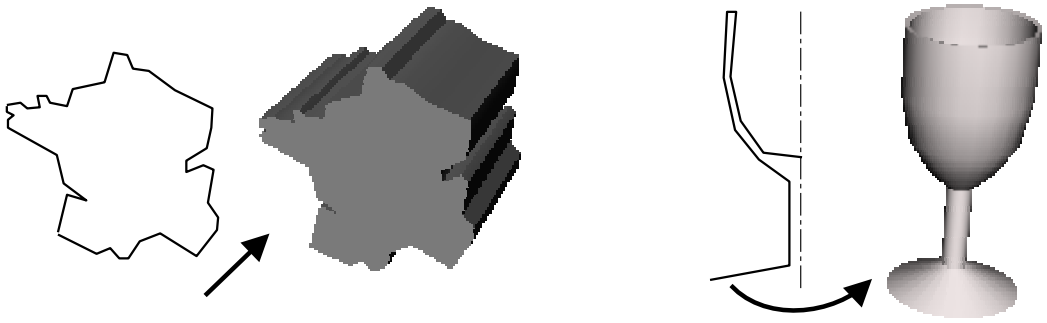

FIG. 1.3 – *Extrusion et outil de révolution*

Les plus classiques sont l'extrusion et l'outil de révolution (figure 1.3). L'extrusion permet du donner une épaisseur à une forme 2D (définie par son contour). L'outil de révolution permet de créer des objets de révolution en faisant tourner un profil autour d'un axe.

Les cylindres généralisés offrent bien plus de possibilités que simplement l'extrusion et la révolution. Pour pouvoir faire glisser un contour le long d'une courbe, il faut associer un repère glissant le long de la courbe. Pour certaines de ses

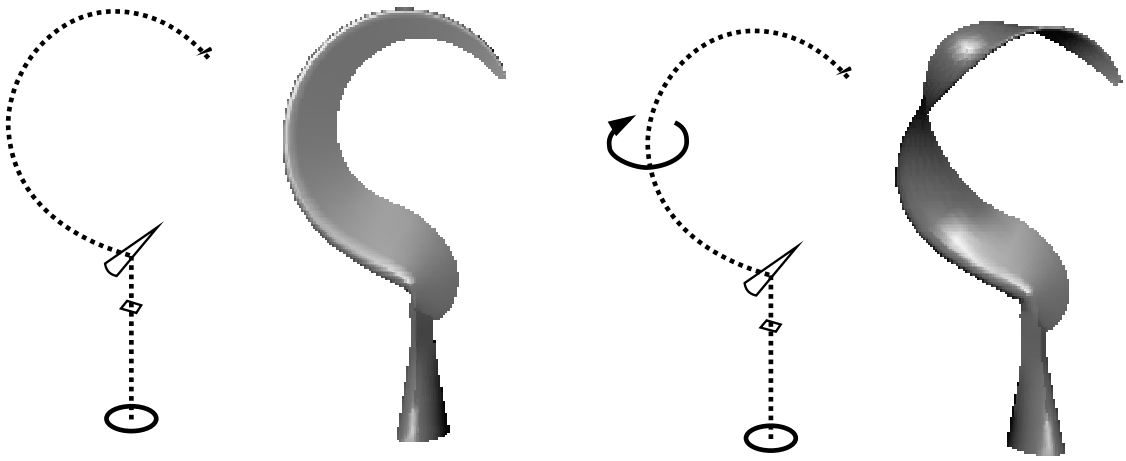

FIG. 1.4 – *Cylindre généralisé obtenu en interpolant plusieurs contours le long d'une courbe (plus une torsion à droite)*

positions, on peut effectuer une transformation sur le repère (changements d'échelle, rotations,...) pour obtenir divers effets sur le résultat final (rétreints, torsions,...) . On peut aussi faire évoluer la forme du contour le long de la courbe, en interpolant entre plusieurs contours par exemple (figure 1.4).

1.3.2 Construction par déformations successives

Les méthodes de construction d'objets par déformations successives sont, en général, très appréciées par les utilisateurs. En effet, elles se rapprochent des techniques utilisées traditionnellement par les sculpteurs. A partir d'un objet de base, généralement très simple, une forme plus complexe peut être modeler en modifiant progressivement l'objet initial grâce à divers outils qui simuleront des opérations du type ajout/retrait de matière à l'objet, lissage, étirement, pincement, courbure,... Ces opérations peuvent s'appliquer plus ou moins localement à une partie de l'objet, en fonction des desiderata de l'utilisateur. Ainsi, il peut construire son objet par affinements successifs. C'est une des méthodes les plus efficaces pour obtenir un résultat qui satisfera visuellement l'utilisateur.

A l'origine de ces techniques, on peut citer les travaux de A. Barr [Bar84]. Un objet est déformé en lui appliquant une succession de déformations. La définition de ces déformations est soit globale, soit locale. Dans le premier cas, une formule mathématique définie explicitement la transformation à appliquer à chaque point de l'objet. Dans le deuxième cas, une formule mathématique définie comment les

vecteurs tangents ou normaux à la surface de l'objet sont transformés.

Il existe de nombreuses techniques pour déformer un objet. La plus simple est celle qui consiste à permettre à l'utilisateur d'agir interactivement sur les paramètres intrinsèques à l'outil qui a permis de construire l'objet ou aux paramètres définissant l'objet (sa représentation). Par exemple, pour un objet construit à partir de carreaux type Bézier, B-spline ou NURBS, l'utilisateur pourra agir sur les points de contrôles de la surface.

L'approche développée par D. Forsey et R. Bartels dans [FB88] utilise les B-splines pour définir la surface de l'objet à modeler. Ces B-splines sont organisées hiérarchiquement de telle sorte qu'un carreau B-spline peut être subdivisé récursivement en quatre carreaux, afin de rajouter des points de contrôle supplémentaires qui permettront à l'utilisateur d'affiner localement l'objet (voir figure 1.5).

FIG. 1.5 – *Modèle construit par B-spline hiérarchique (figure extraite de la page web de D. Forsey* `http://www.cs.ubc.ca/nest/imager/contributions/forsey/dragon/hbsplines.html`*)*

- Les outils de déformation indépendants de la représentation de l'objet

Par opposition aux techniques de déformation précédemment citées qui dépendent de la représentation utilisée, il est possible de définir des outils de déformation qui peuvent s'appliquer à tout type d'objets indépendemment de la représentation utilisée. Les intérêts de ces techniques sont multiples, outre leur relative simplicité, on peut noter qu'il est généralement facile de les intégrer dans les modeleurs existants.

Ces techniques permettent de définir des déformations dites de "forme libre". L'utilisateur contrôle la position, la taille et la frontière de la zone déformée ainsi que la forme de la déformation elle-même.

Le principe de ces déformations consiste à appliquer une transformation mathématique, appelée *déformation de l'espace*, qui va déformer l'espace ou une

zone de l'espace dans lequel est plongé l'objet [Bec94]. L'objet sera alors déformé en conséquence. Ainsi, les FFD (Free-Form Deformations), introduites par T. Sederberg et S. Parry [SP86] et étendues par S. Coquillart [Coq90, CJ91], utilisent un treillis 3D auquel sera associé un polynôme de Berstein de degré 3. La déformation s'effectue en déplaçant les points de contrôle du treillis (figure 1.6).

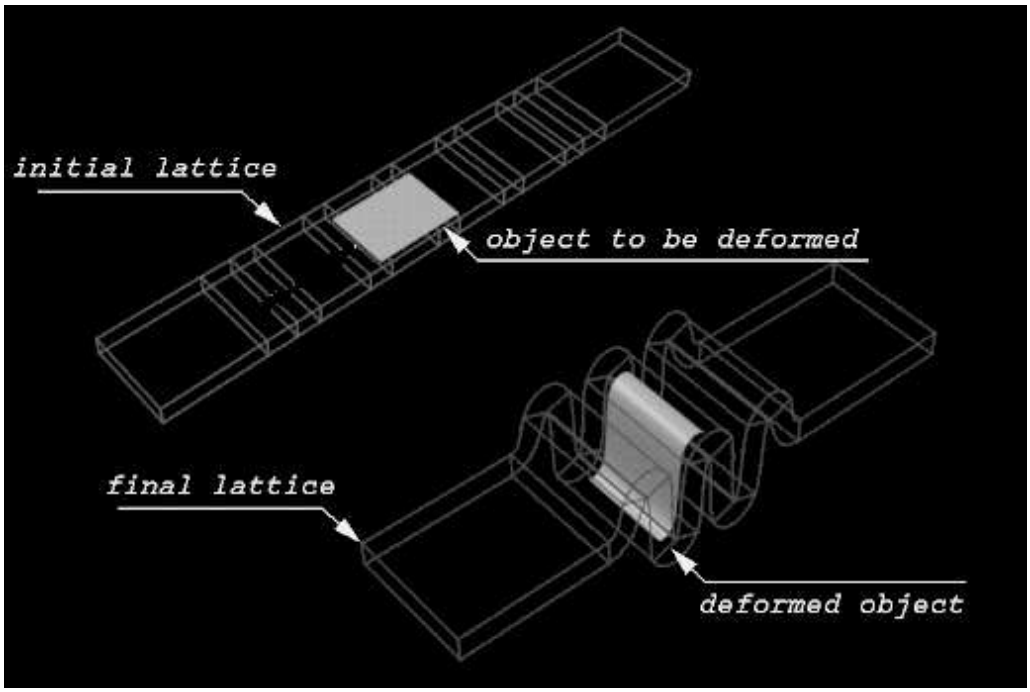

FIG. 1.6 – *Déformation de type FFD (figure fournie par S. Coquillart)*

Une autre façon de voir une déformation est de la considérer comme un champ de déplacement. Utilisant cette idée, P. Borrel et D. Bechmann [BB91] ont développé un outil basé sur l'interpolation de déplacement pour un objet de dimension quelconque plongé dans un espace de dimension n quelconque. Le but est que l'utilisateur n'ait à spécifier que le déplacement d'un nombre restreint de points de l'objet. Le logiciel doit ensuite en déduire le déplacement de tous les points.

Les déformations axiales [LCJ93] associent un axe à l'objet à déformer. L'utilisateur déforme l'axe (en déplaçant ses points de contrôles s'il s'agit d'une spline), et l'objet est alors déformé en conséquence. De plus, un repère local est associé à l'axe; en tournant ce repère le long de l'axe, on peut "vriller" l'objet (figure 1.7).

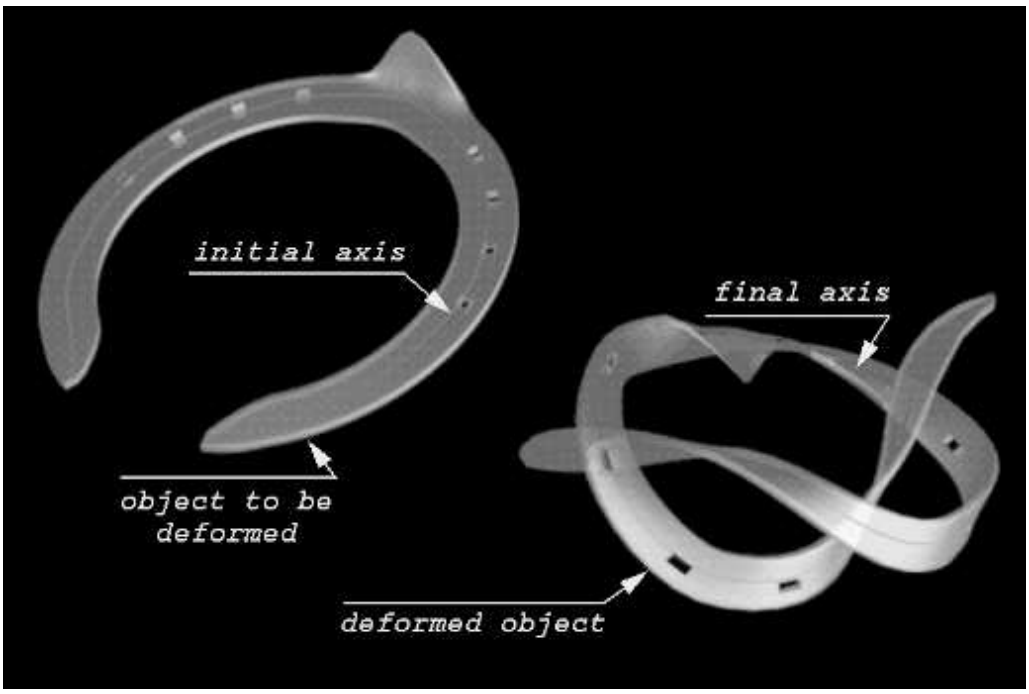

FIG. 1.7 – *Déformation axiale (figure extraite de [LCJ93])*

1.3.3 Construction par combinaison de formes

Les techniques de construction par combinaison de formes 3D se basent sur un algorithme qui permet de créer une nouvelle forme 3D à partir de deux (ou plus) formes données. En général, ce processus est complètement automatique, l'utilisateur se contente de désigner les deux objets de départ.

1.3.3.1 CSG

La combinaison des objets 3D peut être régie par un arbre binaire dont les feuilles sont les objets 3D initiaux et les nœuds sont des opérateurs ensemblistes (union, intersection, différence,...). Les objets initiaux sont, le plus souvent, des primitives très simples : sphère, cône, cylindre, boîte,... . Ce mode de construction, appelé CSG (Constructive Solid Geometry), est souvent utilisé lorsqu'il s'agit de modeler des objets architecturaux ou manufacturés. La figure 1.8 illustre cette technique.

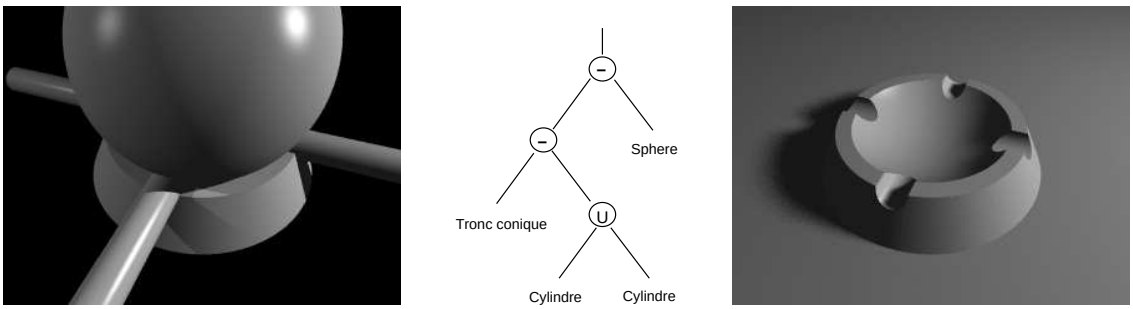

FIG. 1.8 – *Construction d'un cendrier en CSG*

1.3.3.2 Sommes de Minkowski

L'utilisation des sommes de Minkowski permet aussi de combiner deux formes afin d'en obtenir une troisième.

La somme de Minkowski de deux ensembles A et B d'un espace vectoriel est l'ensemble

$$A \oplus B = \{a + b \mid a \in A \text{ et } b \in B\} \quad .$$

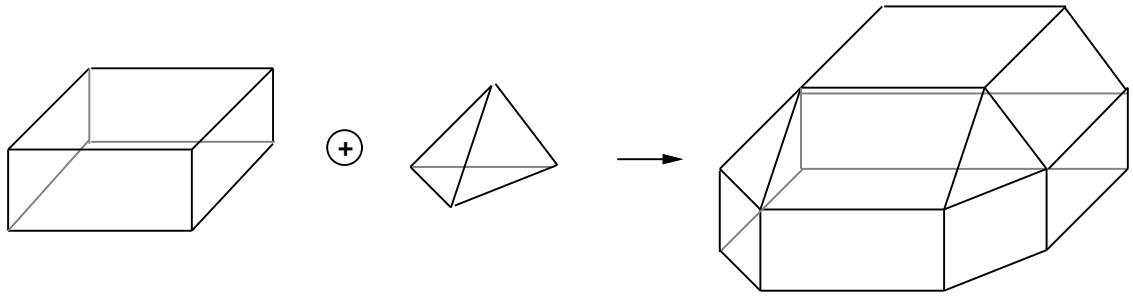

FIG. 1.9 – *Somme de Minkowski de deux polyèdres*

Le calcul de la somme de Minkowski de deux ensembles A et B peut s'effectuer en considérant simplement les frontières des domaines. Ceci permet de construire directement la surface d'un objet issu de la somme de deux autres objets à partir de leur description surfacique uniquement. L'utilisation de cette somme est donc particulièrement bien adaptée à ce type de représentation.

Un autre avantage de la somme de Minkowski est qu'elle peut s'appliquer indifféremment à des objets de topologie quelconque. Elle est donc un outil particulièrement intéressant dans le cadre des métamorphoses d'objets, comme nous allons le voir plus loin.

1.3.3.3 Assemblage de surfaces splines

Pour modéliser un objet à partir de surfaces splines, M-P. Gascuel [Gas89, Gas90] propose un ensemble d'outils permettant d'assembler des morceaux de surface. Les outils contrôlent automatiquement les déformations locales nécessaires en agissant sur les points de contrôle. Le "recollement" positionne et ajuste automatiquement les bords à coller de manière à créer un seul objet sans perte de régularité ; le "pincement" permet de réaliser des embranchements multiples de surfaces tubulaire ; "l'ajustement perpendiculaire" règle le cas où une jonction doit être réalisée perpendiculairement à une surface support ; "l'habillage automatique de squelettes" combine les techniques précédentes pour habiller d'une peau régulière le squelette de l'objet.

1.3.3.4 Surfaces implicites

Contrairement aux méthodes de type CSG qui génèrent des arêtes saillantes au niveau des intersections entre les objets que l'on compose, les techniques de fusion vont chercher à obtenir des jonctions le plus lisse possible entre les objets composés. L'exemple typique est celui de deux bulles de mercure qui se rencontrent pour ne plus former qu'une seule bulle plus grosse.

La modélisation implicite [Bli82, NHTa85, WMW86b] est souvent utilisée pour obtenir ce genre d'effet. Les formes à fusionner sont décrites par une fonction f définissant implicitement leur surface .L'ensemble des points M de l'espace vérifiant $f(M) = Constante$ appartiennent à la surface de l'objet. La fusion de n formes définies par les fonctions $f_1, f_2, \cdots, f_n$ est alors définie par la fonction $F = \sum_i f_i$.

Un objet implicite, d'apparence complexe, est obtenu en sommant les fonctions associées à des objets plus simples. Pour créer une surface implicite autour d'un squelette, on somme les fonctions f_i associées à chaque partie du squelette. Chaque f_i est obtenue en combinant une fonction de distance à l'élément du squelette $d_i(M)$ et une fonction potentielle $g_i(r)$: $f_i(M) = g_i(d_i(M))$.

La représentation d'un objet sous forme implicite permet d'effectuer efficacement des traitements qui auraient demandés beaucoup de calculs si l'objet était représenté explicitement (par un maillage de se surface par exemple). C'est le cas du traitement de la collision entre deux objets flexibles [Gas93].

Cette souplesse d'utilisation, ainsi que l'originalité des formes obtenues par cette méthode, en ont fait un outil très populaire. Elle permet, entre autre, de créer des formes organiques. Nous reviendrons plus en détail sur la modélisation

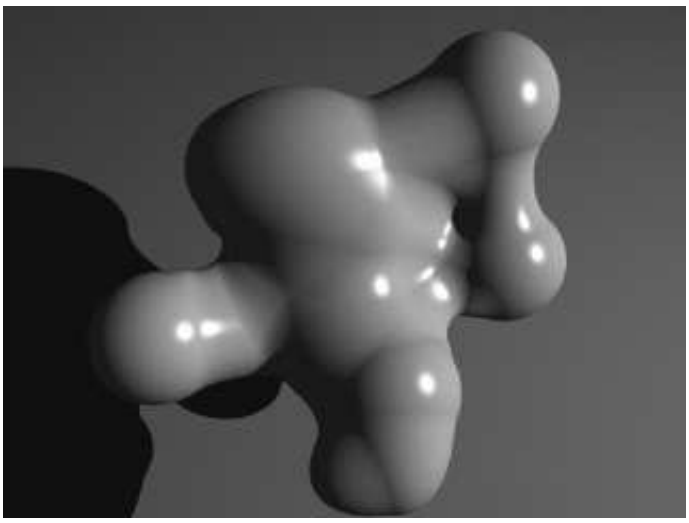

FIG. 1.10 – *Forme générée à partir de dix blobs sphériques (potentiel radial)*

implicite dans la section 1.4 afin de la positionner par rapport à la méthode que nous proposons.

1.3.4 Métamorphoses

La métamorphose d'objets 3D est aussi une façon de combiner deux formes A et B pour en obtenir une troisième C. Cette forme C évolue en fonction d'un paramètre t (généralement assimilé au temps) de telle sorte que C(t=0) = A , C(t=1) = B , et que pour tout t variant de 0 à 1, C(t) se déforme continûment de A vers B.

Il n'existe pas *une* technique de métamorphose, le domaine reste encore très ouvert [KCP92, Hug92, SG92, Laz95, DG96]. La difficulté vient, entre autres, du fait qu'il existe une infinité de façons de métamorphoser un objet en un autre. Celle que l'utilisateur souhaite n'est pas forcement celle que l'algorithme de métamorphose utilisé lui donnera.

On peut tout de même citer une méthode qui a rapport aux techniques de modélisation par combinaison de formes cités plus haut. J. Rossignac et A. Kaul proposent dans [KR91, RK94] d'interpoler deux formes en utilisant les sommes de Minkowski :

$$C(t) = (1-t).A \oplus t.B \quad .$$

Le principal intérêt de cette méthode est qu'elle s'applique aux objets de topologie quelconque puisque c'est une propriété des sommes de Minkowski.

1.4 Application à la modélisation de formes organiques

La modélisation d'objets dont la forme ressemble à une forme organique (comme les corps humains et les corps d'animaux) requiert un modèle capable de décrire et d'animer les formes "douces" et "lisses" des structures organiques, en particulier au niveau des articulations et de l'effet des muscles. Nous allons évoquer deux approches qui permettent de traiter ce problème. La première se base sur des modèles explicites (polygonal et spline), la seconde utilise les modèles implicites.

1.4.1 Par modèles polygonaux

L'utilisation de modèles de type polygonal nécessite l'emploi d'outils intermédiaires pour gérer la déformation de la surface au niveau des muscles et des articulations. Il faut, en effet, pouvoir déplacer les points du maillage de façon cohérente lors de leur animation. Ceci ne peut se faire manuellement car il faut un nombre important de points pour approximer la surface.

J. Chadwick [CHP89] utilise les FFD associés à un squelette. Ainsi, pour modéliser un personnage, il part de son squelette et de la surface représentant l'apparence du personnage (la peau). Les boites de déformation FFD (le treillis) sont placées autour des membres à déformer et à animer. Lorsqu'on agit sur les articulations du squelette, les points de contrôle des boites de déformation sont déplacés et la surface du personnage se déforme alors en conséquence.

1.4.2 Par modèles splines

Dans le cas de l'utilisation de modèles de type spline, il est aussi possible de contrôler à haut niveau, par l'intermédiaire d'un squelette, les déformations dues aux muscles et aux articultions.

Ainsi, D. Forsey [For91] a étendu sa technique de modélisation par B-spline hiérarchique (voir section 1.3.2) dans ce sens. Les points de contrôle des B-splines sont attachés à un squelette qui sera utilisé pour déformer et animer le modèle. Un point de contrôle influe non seulement sur le carreau spline auquel il correspond, mais aussi sur tous les "sous-carreaux" issus de la subdivision de ce carreau. Ainsi, en modifiant ce point de contrôle, la surface est modifiée plus ou moins localement en fonction du niveau hiérarchique du carreau correspondant. Pour obtenir des déformations acceptables au niveau des articluations et des muscles lors

de l'animation, l'utilisateur doit associer judicieusement certains points de contrôles au squelette.

1.4.3 Par modèles implicites

Lorsque l'utilisateur souhaite construire un modèle à partir de rien, il a souvent recourt aux techniques utilisant les modèles implicites car elles permettent d'obtenir des formes de type organiques convainquantes de façon particulièrement intuitive [Gra93]. Cela vient principalement du fait qu'un objet implicite d'apparence relativement complexe peut être généré à partir d'un nombre réduit de primitives de haut niveau. Typiquement, on utilise des metaballs (ellipsoïdes) ; les représentations implicites sous-jacentes seront cumulées pour obtenir la fonction implicite du modèle final. Cette technique a malgré tout quelques inconvénients.

Ainsi, lorsque deux parties d'un même objet sont proches l'une de l'autre, elle vont se rejoindre bien que le modéliste ne le souhaite pas. Par exemple, deux doigts d'une main modelée implicitement peuvent se combiner pour ne former plus qu'un, s'ils sont trop proches. A. Opalach et S. Maddock ont proposé une solution à ce problème [OM93] en utilisant graphe qui spécifie quelles sont les primitives qui peuvent se combiner entre elles.

Un autre inconvénient vient du fait que le rendu des objets implicites est, en général, coûteux. La polygonalisation ou le rendu direct en lancer de rayon de l'objet sont difficilement obtenus en temps interactif. Par conséquent, la manipulation des modèles implicites dans les modeleurs 3D interactifs n'est pas facile. Un algorithme de polygonalisation rapide est proposé dans [DTG95] afin de permettre le modelage de tels objets avec visualisation interactive.

Et enfin, la texturation des objets implicites est un problème difficile [Ped95]. Les méthodes de type *plaquage de texture* ne sont pas utilisables directement car il faudrait pouvoir disposer d'un paramétrage de leur surface. On peut, par contre, polygonaliser la surface de l'objet implicite et associer des coordonnées textures aux sommets des polygones (méthode utilisée, par exemple, dans les logiciels de peinture directe sur la surface de l'objet) ; mais le problème n'est pas complètement résolu car il faut aussi que la texture reste fixée à la surface de l'objet de manière cohérente lorsque celui-ci se déforme au cours d'une animation. Très récemment, J-P. Smets-Solanes [SS96] a proposé une solution pour maintenir cette cohérence. Une "peau virtuelle" surlaquelle est plaquée la texture est associée à l'objet. Un champ de vecteur est utilisé pour maintenir la peau collée de façon cohérente sur l'objet

pendant l'animation de celui-ci.

J. Shen [ST95] propose une méthode combinant les modèles implicites et les splines : la surface est caractérisée implicitement par un ensemble de metaballs ellipsoïdales et est approximée par des carreaux de surface spline. La surface implicite est échantillonnée radialement par rapport à un axe spécifié par l'utilisateur ; les points issus de cet échantillonnage servent de points de contrôle des splines. Le temps de calcul est faible. Ainsi, il est possible de créer interactivement des personnages articulés et de les texturer convenablement.

Dans la suite, nous proposons une autre approche, certes plus limitée que la modélisation implicite, mais qui permet de garder sa facilité d'utilisation, sans en avoir les inconvénients cités ci-dessus. Elle est basée sur une technique de fusion d'objets décrite dans le chapitre suivant.

Chapitre 2

La fusion d'objets

Dans le cadre de la modélisation par combinaison de formes, nous avons développé, dans [Dec93, DG94], une méthode originale permettant de fusionner deux objets définis explicitement par un maillage de leur surface.

2.1 Principe

Partant de deux objets A et B donnés qui s'intersectent, cette méthode calcule un troisième objet C qui respecte les contraintes suivantes :

- C englobe A et B (tout point inclus dans A ou dans B est inclus dans C),
- la surface de C est "la plus lisse possible" [1],
- Le volume de C est égal à la somme des volumes de A et de B (il y a conservation du volume).

L'idée est de considérer qu'il y a un surplus de matière au niveau de l'intersection entre A et B : si on considère que A et B sont remplis d'une matière de

1. Si les surfaces de A et de B sont $\mathcal{C}^n$ (continûment différentiables à l'ordre n) alors la surface de C est aussi $\mathcal{C}^n$

densité homogène, alors la densité au niveau de A ∩ B est double. On peut considérer qu'il s'agit là d'un surplus de matière (correspondant au volume de A ∩ B). L'idée est de reporter ce surplus de matière vers l'extérieur de A ∪ B. Reste à définir la stratégie permettant de reporter cette matière.

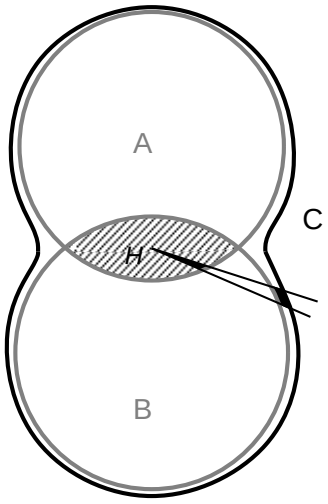

FIG. 2.1 – *Principe de la fusion (en 2D) : le surplus de matière est reporté radialement vers l'extérieur*

2.2 Cas des objets étoilés

Dans le cas d'objets étoilés[2] (figures 2.3 et 2.4) par rapport à un même point H (inclus dans A ∩ B). Le surplus de matière est alors reporté radialement vers l'extérieur par rapport à ce point H (voir figure 2.1) : pour chaque secteur angulaire de section infinitésimale centré en H, la matière contenue dans l'intersection de ce secteur avec A ∩ B est reportée à l'extrémité de l'intersection entre le secteur et A ∪ B.

La surface de l'objet C peut s'exprimer simplement en utilisant les coordonnées sphériques, H est le centre du système de coordonnées : La surface de A est définie par la fonction $\rho_A(\theta, \varphi)$ et la surface de B est définie par la fonction $\rho_B(\theta, \varphi)$. En posant que pour chaque secteur angulaire $s(\theta, \varphi)$ de section infinitésimale et centré

2. Un objet $\mathcal{A}$ est étoilée par rapport au point H si, pour tout point M à l'intérieur de $\mathcal{A}$, le segment $[HM]$ est inclus dans $\mathcal{A}$.

en H, le volume de C $\cap$ $s(\theta,\varphi)$ est égal à la somme des volumes de A $\cap$ $s(\theta,\varphi)$ et de B $\cap$ $s(\theta,\varphi)$, on obtient la fonction polaire $\rho_C(\theta,\varphi)$ définissant la surface de C :

$$\rho_C(\theta,\varphi) = \sqrt[3]{\rho_A(\theta,\varphi)^3 + \rho_B(\theta,\varphi)^3} \tag{2.1}$$

Ceci sera démontré dans le chapitre suivant (section 3.1.2.2).

Pour une direction $\vec{u}$ donnée ($\vec{u}$ est un vecteur normé), on peut calculer un point M de la surface de C de la façon suivante :

$$M = H + \left(\sqrt[3]{HI^3 + HJ^3}\right)\vec{u}$$

où I (resp. J) est l'intersection de la demi-droite partant de H et de direction $\vec{u}$ avec la surface de A (resp. B).

La technique de déformation par fusion que nous allons décrire dans la suite de cette thèse est une extension directe de celle-ci. Nous aurons donc l'occasion d'y revenir plus en détail.

2.3 Cas des objets de formes quelconques

La fusion d'objets de formes quelconques par cette méthode est plus difficile. En effet, cela suppose de régler deux problèmes non triviaux :

- Où et comment reporter le surplus de matière ? L'intersection d'un secteur angulaire avec A $\cup$ B n'est pas unique (voir figure 2.2).
- Comment gérer les changements de topologie ? Si A et B ont une forme et une topologie quelconque, C peut avoir une topologie différente, issue de l'union de A et B.

Ces problèmes restent ouverts, mais des débuts de solution sont proposés dans [Dec93], [DG94] et [Bal96]. Ainsi, le problème du report de la matière peut se faire en considérant, pour chaque secteur angulaire, toutes les zones "vides" de ce secteur, c'est-à-dire n'intersectant ni A ni B. On commence alors à reporter le surplus de matière dans la zone vide la plus proche de H. Lorsque cette zone est remplie et qu'il reste encore de la matière à reporter, elle est reportée dans la zone suivante, et ainsi de suite.

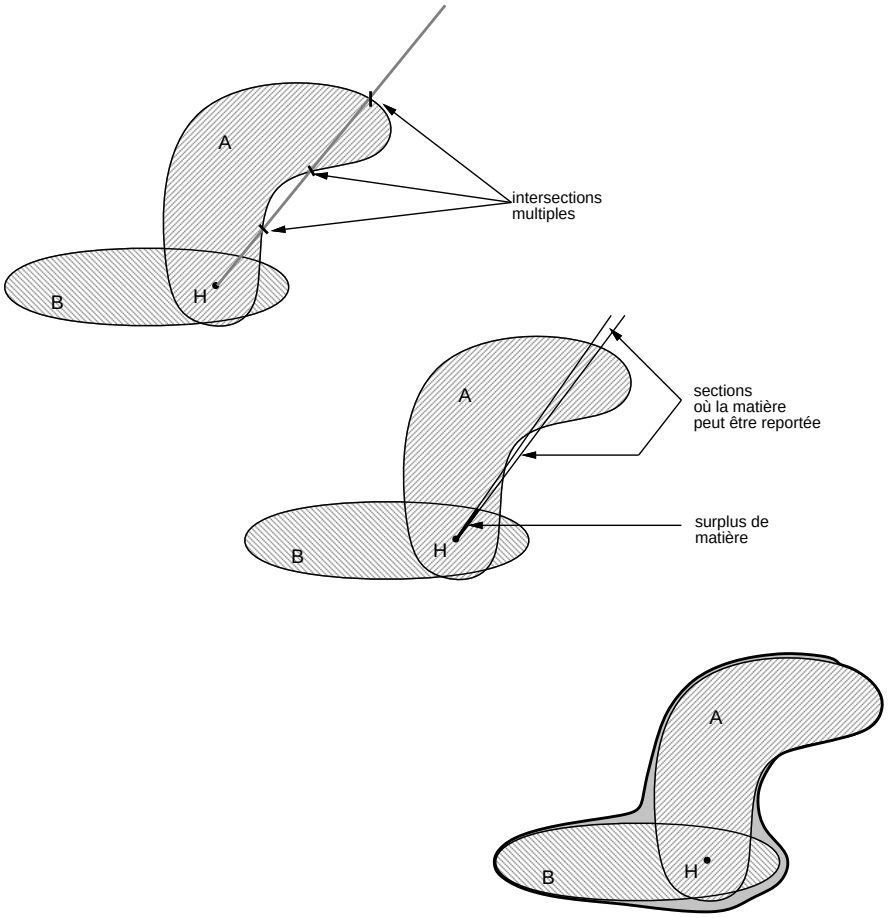

FIG. 2.2 – *Problème du report de la matière dans le cas d'objets quelconques*

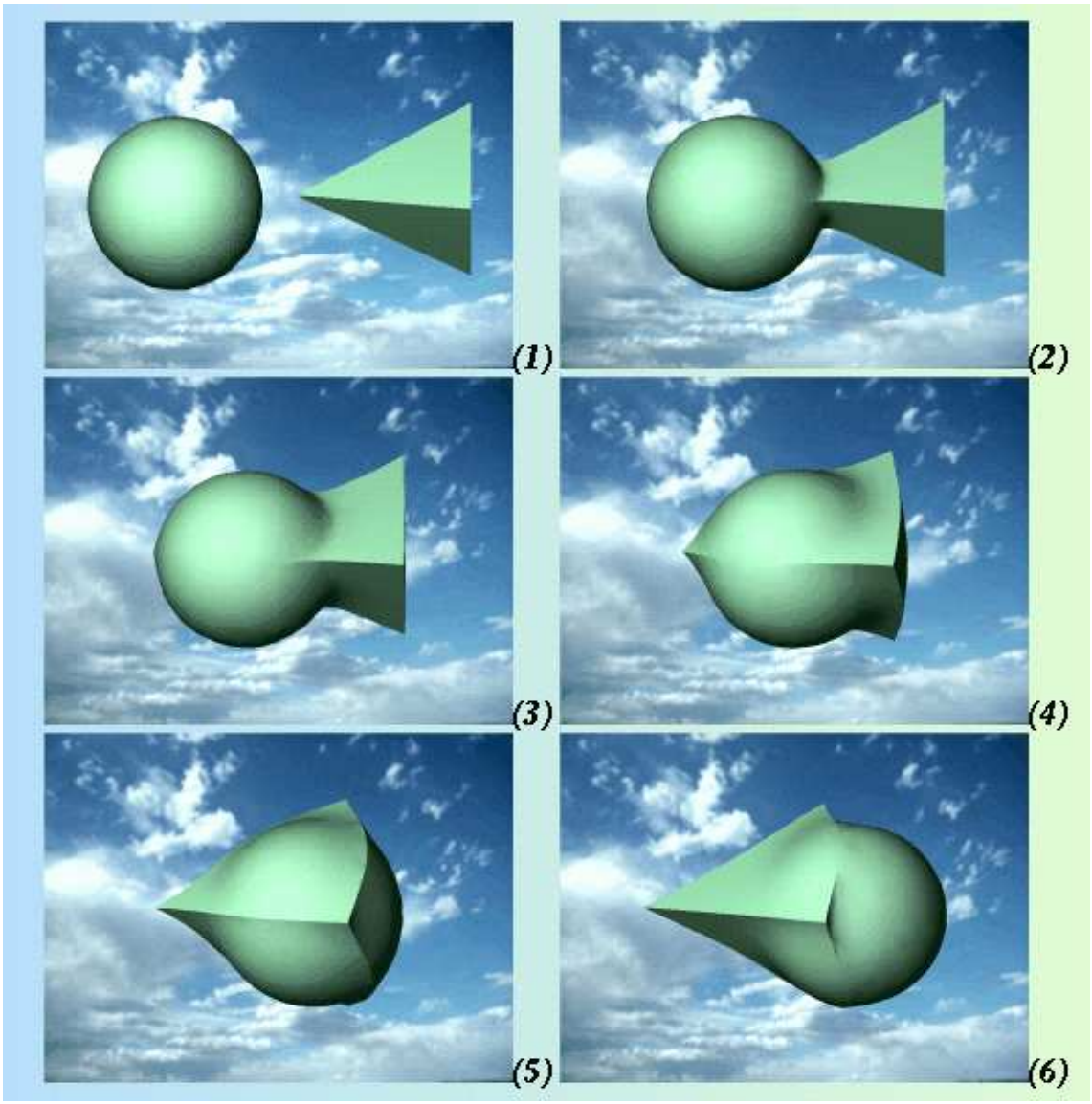

FIG. 2.3 – *Fusion d'une sphère et d'un tétraèdre*

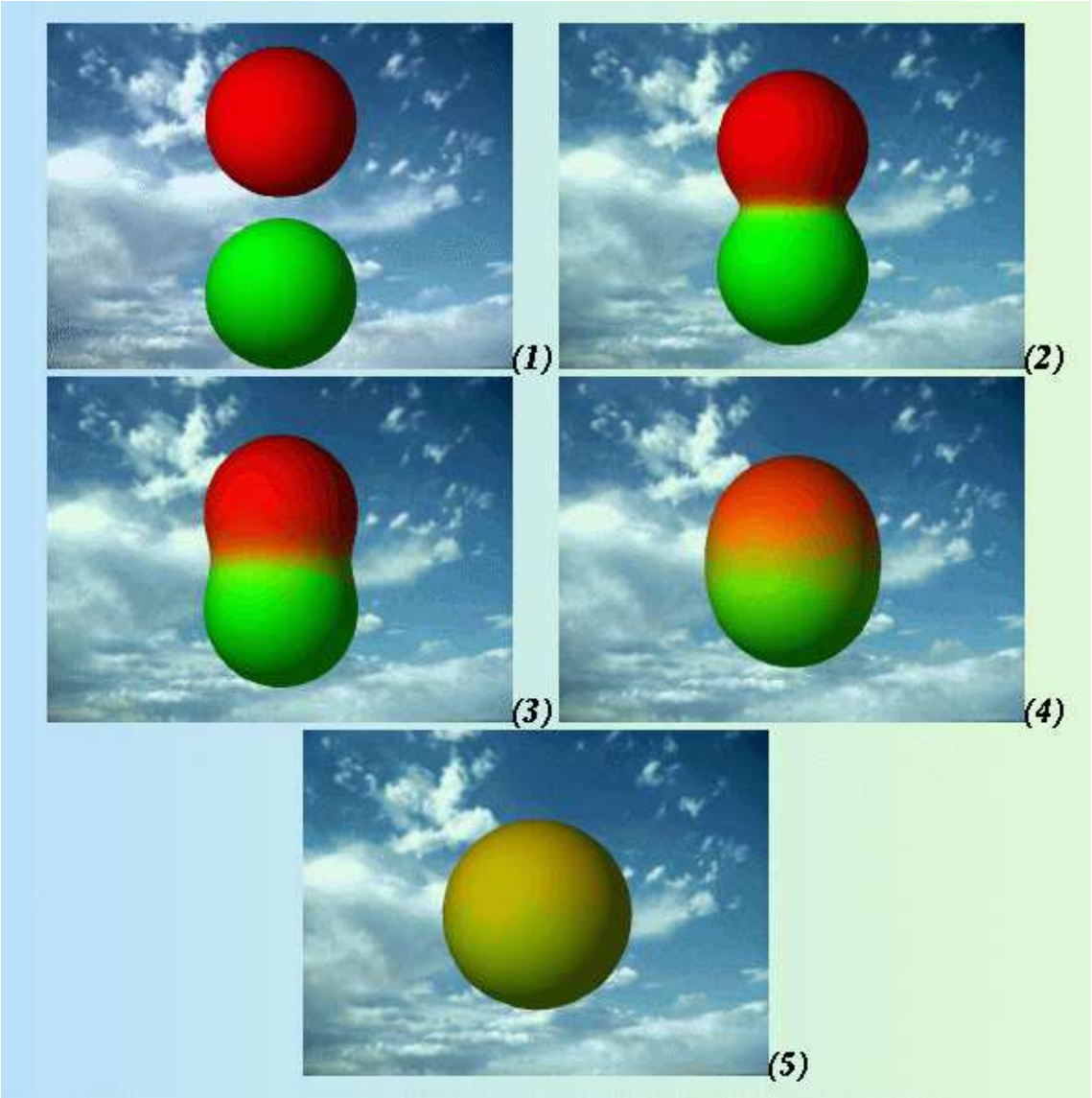

FIG. 2.4 – *Fusion de deux bulles de couleur*

2.4 Métamorphoses basées sur cette méthode

Dans [DG94], nous avons utilisé une méthode similaire à celle utilisée par J. Rossignac et A. Kaul dans [KR91] : la transformation d'un objet A en un objet B y est obtenue en calculant à chaque instant l'objet C tel que $\mathrm{C}(t) = (1-t).\mathrm{A} \oplus t.\mathrm{B}$. Ainsi, pour métamorphoser des objets étoilés, nous utilisons notre technique de fusion d'objets (section 2.1) à la place des sommes de Minkowski. Un des intérêts de cette méthode est de garder le contrôle du volume au cours de la métamorphose : l'objet A est fusionné avec l'objet B en diminuant le volume de A de 100% à 0% et en augmentant le volume de B de 0% à 100%. On peut exprimer ceci de manière plus formelle par :

$$\mathrm{C}(t) = f_H(h_H(\sqrt[3]{1-t}, \mathrm{A}), h_H(\sqrt[3]{t}, \mathrm{B}))$$

où H est un point choisi[3] dans $\mathrm{A} \cap \mathrm{B}$, $h_H(\alpha, F)$ représente l'homothétie par rapport à H de rapport α appliquée à l'objet F, et $f_H(F, G)$ représente la fusion par rapport à H des objets F et G. Comme $volume(h_H(\alpha, F)) = \alpha^3 \times volume(F)$ et $volume(f_H(F, G)) = volume(F) + volume(G)$ car la fusion conserve le volume, on a donc

$$volume(\mathrm{C}(t)) = (1-t) \times volume(\mathrm{A}) + t \times volume(\mathrm{B})$$

cela signifie que l'on passe linéairement du volume de A au volume de B lors de la métamorphose.

L'intérêt de la méthode reste malgré tout limité puisque restreint à des objets étoilés pour lesquels les méthodes de type interpolation [KCP92] donnent aussi des résultats convainquants. L'avantage principal que l'on peut y voir est de garder le contrôle du volume de l'objet au cours de la transformation.

2.5 Vers un outil de déformation

La technique de fusion a été présentée ici comme une technique de composition d'objets. Partant de deux objets, un troisième objet est créé. Il est aussi possible de considérer cette technique comme un outil de déformation. Un objet est déformé par un autre ; ce deuxième objet est considéré comme un outil. La déformation est obtenue en fusionnant l'objet et l'outil.

Dans ce cadre, il est souhaitable que l'outil de déformation puisse s'appliquer aux objets de formes quelconques. Le but est de diminuer le plus possible les

3. on peut choisir H comme étant le barycentre de $\mathrm{A} \cap \mathrm{B}$ par exemple

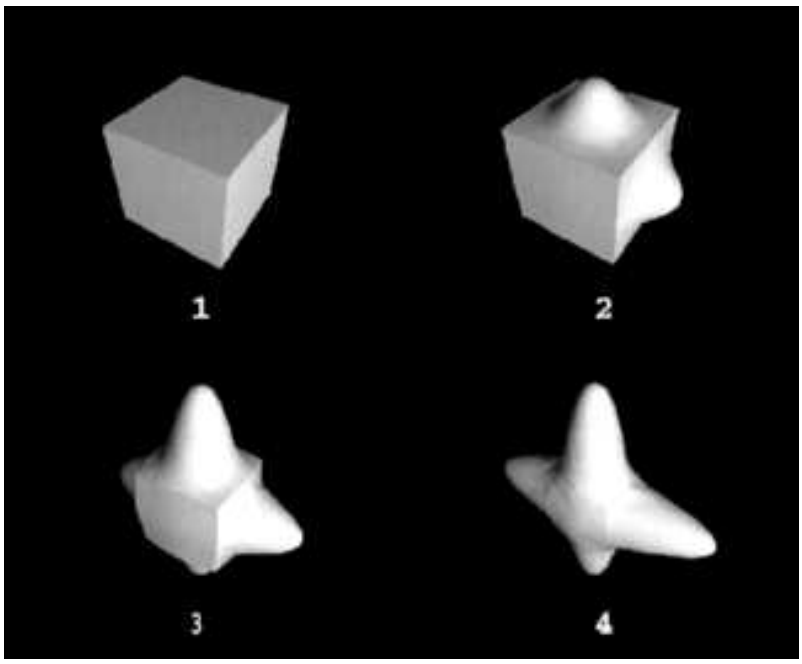

FIG. 2.5 – *Métamorphose d'un cube en étoile par fusion d'objets*

contraintes imposées à l'utilisateur de cet outil. Par contre, la forme de l'outil peut rester simple (étoilée ou simplement convexe). Comme nous le verrons dans la suite, des déformations complexes peuvent être obtenues en appliquant successivement plusieurs déformations simples à un objet.

Le chapitre suivant présente l'outil de déformation de type fusion. Pour cela, l'équation 2.1 est utilisée pour définir une fonction de déformation de l'espace. Un objet est déformé en appliquant cette transformation à tous les points de sa surface. La question du "report de matière" ne se pose plus. La déformation s'applique alors à tout type d'objet ; seule la forme de l'outil est limitée. On ne peut plus considérer qu'il s'agit d'une "fusion" comme nous venons de la définir car la contrainte d'inclusion n'est plus forcement vérifiée : l'objet déformé n'inclus pas toujours l'objet initial et l'outil. Par contre, la contriante de continuité et la conservation du volume sont vérifiées.

Chapitre 3

Outils de déformation

Ce chapitre présente les deux outils de déformation d'espace que nous avons développés :

- la déformation par fusion,
- la déformation par flexion.

Les effets de déformation obtenus permettent d'étendre la palette des outils de type déformation de l'espace mise à la disposition de l'utilisateur. Ils sont aussi à la base du modèle dédié à la modélisation de formes organiques que nous proposerons dans le chapitre suivant.

3.1 Déformation par fusion d'un objet avec une forme 3D simple

3.1.1 Aperçu

La déformation par fusion est une méthode de type *déformation de l'espace*. Une transformation mathématique est appliquée à l'espace 3D dans lequel repose l'objet, l'objet sera alors déformé en conséquence.

L'outil de déformation regroupe l'ensemble des paramètres qui définissent la transformation. Il est composé d'une forme 3D convexe positionné dans l'espace (appelé *forme de l'outil*) et d'un point 3D inclus dans la forme (appelé *centre de l'outil*). Cet outil peut déformer un objet de deux façons :

- si le centre de l'outil est à l'intérieur de l'objet, l'outil produit alors une bosse sur l'objet de telle façon que l'objet déformé englobe la forme de l'outil, une sorte de fusion entre l'objet et la forme s'opère (cf. figure 3.1) ;
- si le centre de l'outil est à l'extérieur de l'objet, l'outil produit alors un creux dans l'objet ; en fait, le complémentaire de l'objet est fusionné avec l'outil (cf. figure 3.2).

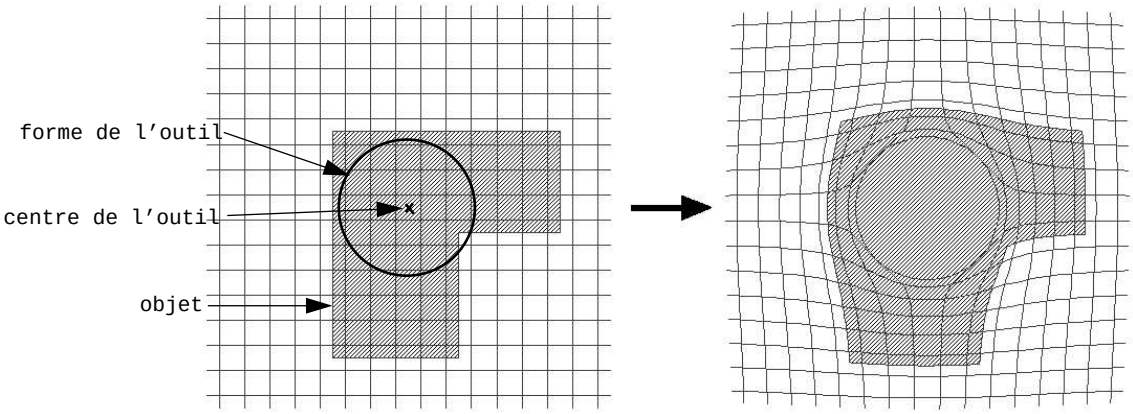

FIG. 3.1 – *L'outil produit une bosse sur l'objet*

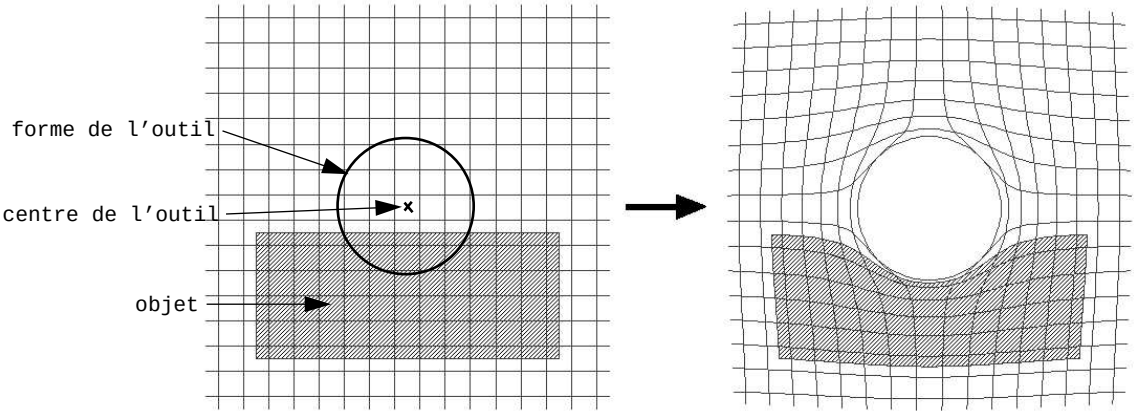

FIG. 3.2 – *L'outil produit un creux sur l'objet*

Cette technique est intéressante car :

- Elle est interactive. Les coûts de calcul sont très faibles, ce qui permet de calculer les déformations en temps interactif sur des stations de travail standards (l'algorithme a été implémenté en C++ sur un station Silicon Graphics Indigo^2) et l'utilisateur peut voir quasi-instantanément le résultat de la modification des attributs de l'outil.

- L'utilisation de l'outil est intuitive. On peut imaginer que l'objet est fait d'une matière élastique (telle que du caoutchouc) et qu'un ballon, dont la forme est celle de l'outil, est gonflé à l'intérieur ou à l'extérieur de l'objet (en fonction de la position du centre de l'outil). L'objet subit alors une déformation en creux ou en bosse correspondant à la forme de l'outil. Ainsi, l'utilisateur peut facilement deviner quel sera l'effet obtenu s'il applique une telle déformation ou s'il en modifie les paramètres.

- Cette technique ne dépend pas du modèle utilisé pour représenter l'objet. En particulier, les objets décrits en facettes (maillage polygonal de leur surface), souvent utilisés dans les modeleurs interactifs, sont bien adaptés. Ce cas sera traité dans la section 3.1.3. Une méthode simple, qui permet de raffiner le maillage pour approximer plus finement l'objet déformé, y est proposé.

- Les déformations obtenues s'approchent de celles que seules les techniques de composition peuvent produire. Prenons l'exemple d'une goutte qui se forme à la surface d'un objet. En utilisant des modèles implicites, on le simulerait en ajoutant un nouveau champ de potentiel de type radial à l'objet, ce qui nécessite, bien sûr, d'avoir une représentation implicite de l'objet. Notre méthode permet d'obtenir cet effet en fusionnant l'objet, décrit explicitement, avec un outil de forme sphérique. La forme de l'outil n'est pas réduite aux formes sphériques, toute forme convexe peut convenir.

3.1.2 La technique de déformation par fusion

3.1.2.1 Description

Pour déformer un objet, nous déformons l'espace 3D (noté $\mathcal{E}$) dans lequel l'objet est placé. La forme de l'outil ($\mathcal{S}$) et le centre de l'outil (H) définissent la déformation. $\mathcal{S}$ est convexe et H est inclus dans $\mathcal{S}$. La déformation s'obtient grâce à une simple application de $\mathcal{E} - \{H\}$ sur $\mathcal{E} - \mathcal{S}$ (un trou est créé dans l'espace

$\mathcal{E}$, voir figure 3.3). Nous utilisons les coordonnées sphériques pour définir cette transformation.

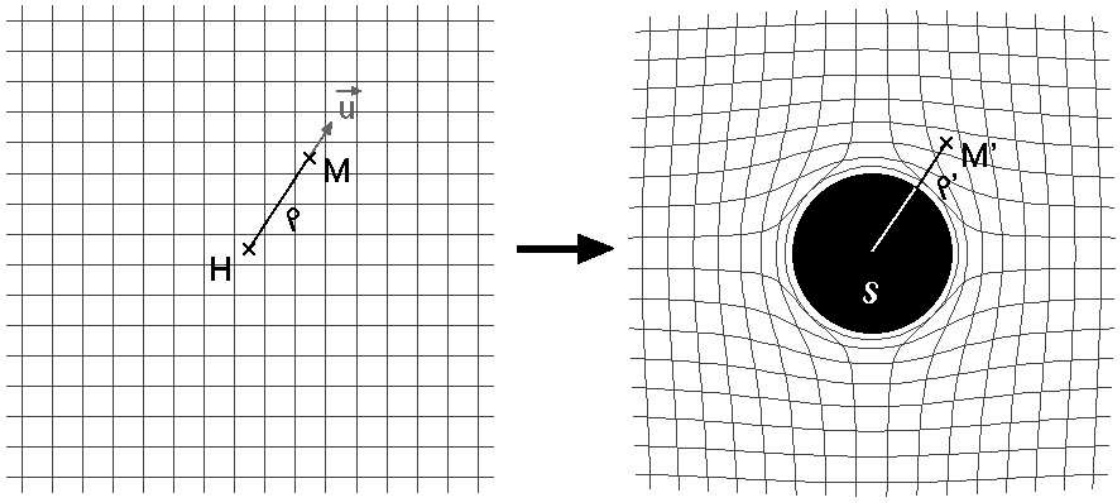

FIG. 3.3 – *Espace déformé par un outil sphérique (en 2D)*

Soit M un point de $\mathcal{E} - \{H\}$,
soit $\vec{u}$ un vecteur normalisé de direction $\overrightarrow{HM}$ (i.e. $\vec{u} = \frac{\overrightarrow{HM}}{\|\overrightarrow{HM}\|}$),
soit ρ la distance entre H et M (i.e. $\rho = \|\overrightarrow{HM}\|$) $\quad \Longrightarrow M = H + \rho\vec{u}$,
soit I l'intersection[1] entre la surface de $\mathcal{S}$ et la demi-droite $(H, \vec{u})$,
soit ρ_0 la distance entre H et I (i.e. $\rho_0 = \|\overrightarrow{HI}\|$) $\quad \Longrightarrow I = H + \rho_0\vec{u}$.
Le point M est transformé en M' de telle sorte que

$$M' = H + \rho'\vec{u} \quad \text{où} \quad \boxed{\rho' = \sqrt[3]{{\rho_0}^3 + \rho^3}} \tag{3.1}$$

Nous avons choisi cette transformation car elle présente des propriétés intéressantes que nous allons détailler.

3.1.2.2 Propriétés

- Continuité

L'équation (3.1) définie une application continue de $\mathcal{E} - \{H\}$ vers $\mathcal{E} - \mathcal{S}$:

Étant donné une surface $\mathcal{C}^n$ (continûment différentiable à l'ordre n), si la surface de la forme de l'outil est $\mathcal{C}^n$, alors la surface déformée est aussi $\mathcal{C}^n$.

1. I est unique car $\mathcal{S}$ est convexe et H à l'intérieur de $\mathcal{S}$.

Cette propriété assure que notre outil génère des déformations qui conservent un aspect *lisse* aux objets.

Preuve :

Nous allons démontrer la continuité en 2D, la démonstration en 3D est à peu de chose près identique, mais les notations sont plus lourdes.

Pour cela, nous allons considérer les frontières de l'objet et de l'outil comme décrites, au moins localement, par les fonctions polaires $\rho(\theta)$ et $\rho_0(\theta)$ et appelons ρ_{fus} la fonction polaire décrivant alors la frontière de l'objet déformé (nous ne l'avons pas appelé ρ' pour qu'il n'y ait pas d'ambiguïté avec la dérivée première de ρ, cette dérivée première sera d'ailleurs notée $\rho^{(1)}$).

Montrons que si ρ et ρ_0 sont continues à l'ordre n ($\mathcal{C}^n$) et si H n'appartient ni à la frontière de l'objet à déformé ni à celle de l'outil (i.e. $\forall\ \theta,\ \rho(\theta) > 0$ ou $\rho_0(\theta) > 0$) alors, ρ_{fus} est $\mathcal{C}^n$.

En 2D, l'équation (3.1) s'écrit :

$$\rho_{fus}(\theta) = \sqrt{\rho(\theta)^2 + \rho_0(\theta)^2} \quad .$$

La justification est fournie dans le paragraphe "conservation du volume" ci-après.

a) Montrons par récurrence que

$$\rho_{fus}{}^{(n)} = \frac{d^n \rho_{fus}}{d\theta^n} = \mathcal{P}_n(\rho, \rho^{(1)}, \cdots, \rho^{(n)}, \rho_0, \rho_0{}^{(1)}, \cdots, \rho_0{}^{(n)})(\rho^2 + \rho_0{}^2)^{\frac{1}{2}-n}$$

où $\mathcal{P}_n$ est un polynôme de degré n à $2(n+1)$ variables.

La propriété est vraie à l'ordre 0. Supposons qu'elle soit vraie à l'ordre $n-1$.

$$\rho_{fus}{}^{(n)} = \frac{d[\rho_{fus}{}^{(n-1)}]}{d\theta} = \frac{d[\mathcal{P}_{n-1}(\rho^2 + \rho_0{}^2)^{\frac{1}{2}-(n-1)}]}{d\theta}$$
$$= \underbrace{\left[(\rho^2 + \rho_0{}^2)\mathcal{P}_{n-1}{}^{(1)} + (\frac{1}{2} - n + 1)(2\rho\rho^{(1)} + 2\rho_0\rho_0{}^{(1)})\mathcal{P}_{n-1}\right]}_{\mathcal{P}_n} (\rho^2 + \rho_0{}^2)^{\frac{1}{2}-n}$$

Donc, $\rho^{(n)} = \mathcal{P}_n(\rho^2 + \rho_0{}^2)^{\frac{1}{2}-n}$ où $\mathcal{P}_n$ est un polynôme de degré n en $(\rho, \rho^{(1)}, \cdots, \rho^{(n)}, \rho_0, \rho_0{}^{(1)}, \cdots, \rho_0{}^{(n)})$. La propriété est vraie à l'ordre 0, donc par récurrence, elle est vraie quelque soit $n \in \mathbb{N}$.

b) $\mathcal{P}_n$ continu car ρ et ρ_0 sont $\mathcal{C}^n$, $(\rho^2 + \rho_0^2)^{\frac{1}{2}-n}$ continu car $\rho^2 + \rho_0{}^2$ ne s'annule pas (H n'appartient ni à la surface de l'objet ni à celle de l'outil par hypothèse). Donc, $\rho^{(n)} = \mathcal{P}_n(\rho^2 + \rho_0{}^2)^{\frac{1}{2}-n}$ et continue.

$\Longrightarrow$ ρ **est** $\mathcal{C}^n$.

- Localité

La déformation s'applique localement à proximité de l'outil $\mathcal{S}$. En effet, si la zone de l'objet à déformer est loin de $\mathcal{S}$, alors la déformation est négligeable.

$$\rho' = \sqrt[3]{{\rho_0}^3 + \rho^3} = \rho\sqrt[3]{1 + \left(\frac{\rho_0}{\rho}\right)^3}$$

$$\text{Si } \rho \gg \rho_0, \text{ alors } \rho' \approx \rho \ .$$

La localité de la déformation est intéressante pour un outil de modélisation car elle permet à l'utilisateur de ne déformer qu'une partie d'un objet sans que le reste de l'objet soit modifié.

- Conservation du volume

Considérons un objet de volume V,

- si le centre de l'outil H est inclus dans l'objet, alors l'objet sera transformé en objet de volume V' tel que $V' = V + V_0$ où V_0 est le volume correspondant à la forme de l'outil $\mathcal{S}$;

- si H n'est pas inclus dans l'objet, alors l'objet est transformé en un objet de même volume.

La conservation du volume est induite par la propriété suivante :

Si on considère un secteur angulaire centré en H et de longueur ρ, lors de la déformation, il est transformé en un secteur angulaire dont le volume a été augmenté par le volume de l'intersection du secteur et $\mathcal{S}$ (cf. figures 3.4 et 3.5).

Preuve :

En 2D, la démonstration est la suivante :

a) Cas où H est à l'intérieur de l'objet à déformer. Soit $\vec{u}(\theta)$ le vecteur $(\cos(\theta), \sin(\theta))$. Soient $\rho_0(\theta)$ et $\rho(\theta)$ les fonctions polaires décrivant respectivement la surface de $\mathcal{S}$ et la première intersection de la demi-droite $[H, \vec{u}(\theta))$ avec la surface de l'objet.

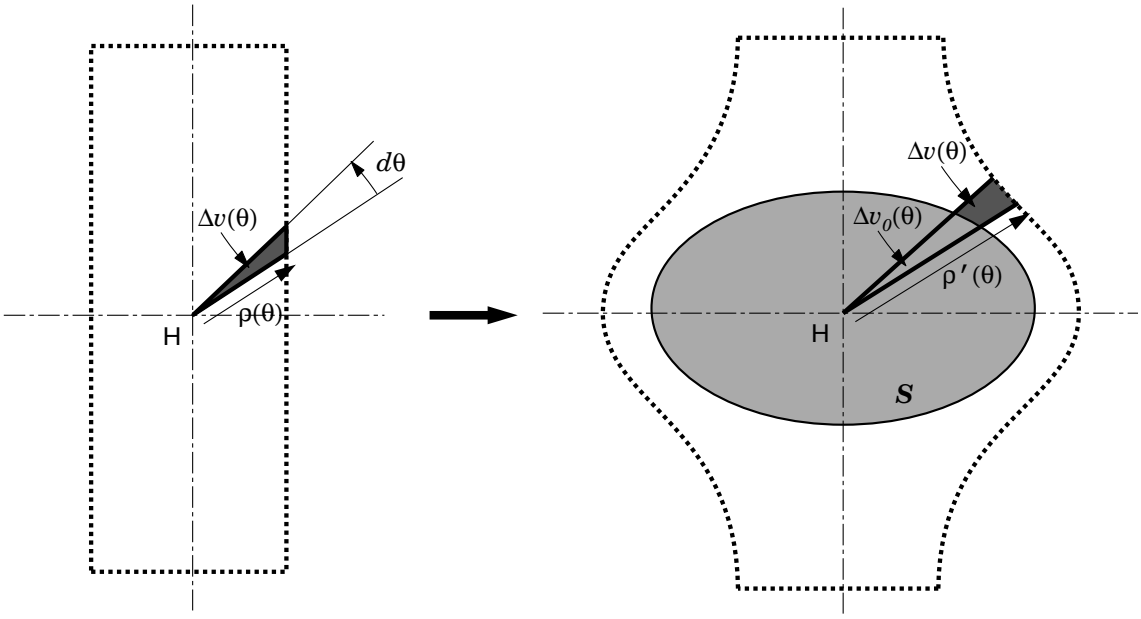

FIG. 3.4 – *Conservation du volume (en 2D). Cas où le centre de l'outil est à l'intérieur de l'objet à déformer*

Soient $\Delta v_0(\theta)$ et $\Delta v(\theta)$ les aires des secteurs angulaires centrés en H et définis respectivement par $(\rho_0(\theta), d\theta)$ et $(\rho(\theta), d\theta)$.

$$\Delta v_0(\theta) = \frac{1}{2}\rho_0(\theta)^2 d\theta$$

$$\Delta v(\theta) = \frac{1}{2}\rho(\theta)^2 d\theta$$

Cherchons une fonction polaire $\rho'(\theta)$, qui correspondra à l'arc déformé, telle que $\Delta v'(\theta)$, l'aire du secteur angulaire définie par $(\rho'(\theta), d\theta)$, satisfasse :

$$\Delta v'(\theta) = \Delta v_0(\theta) + \Delta v(\theta)$$

Cette équation peut être interprétée comme la conservation de l'aire le long d'un secteur angulaire.

$$\Delta v'(\theta) = \frac{1}{2}\rho'(\theta)^2 d\theta$$

$$\frac{1}{2}\rho'(\theta)^2 d\theta = \frac{1}{2}\rho_0(\theta)^2 d\theta + \frac{1}{2}\rho(\theta)^2 d\theta$$

En résolvant cette équation, nous pouvons montrer que :

$$\rho'(\theta) = \sqrt{\rho_0(\theta)^2 + \rho(\theta)^2}$$

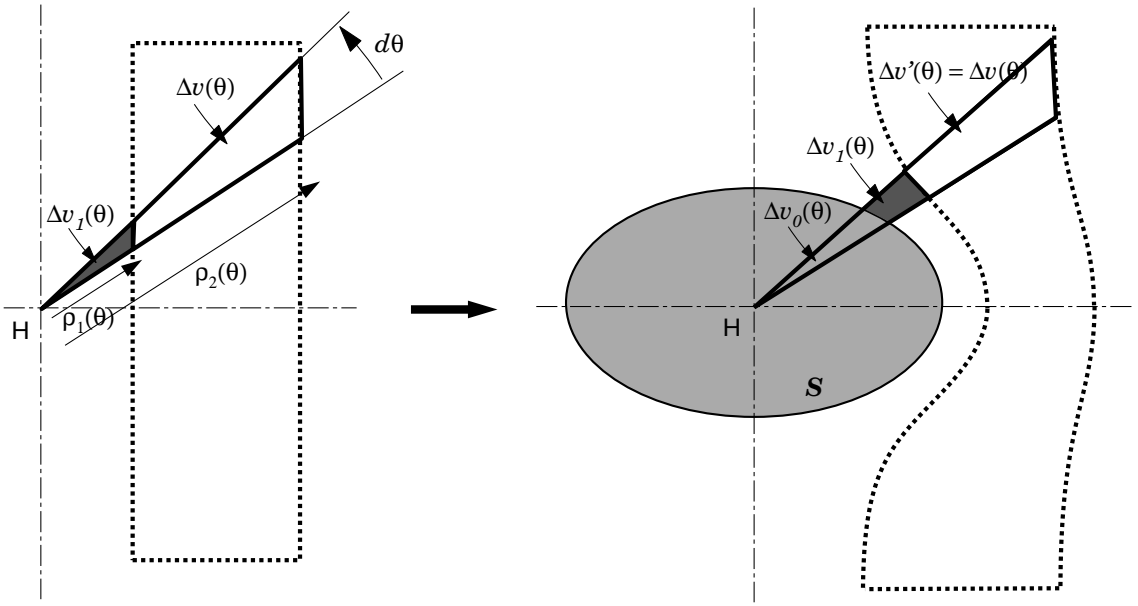

FIG. 3.5 – *Conservation du volume (en 2D). Cas où le centre de l'outil est à l'extérieur de l'objet à déformer*

b) Cas où H est à l'extérieur de l'objet à déformer. Soit $\vec{u}(\theta)$ le vecteur $(\cos(\theta), \sin(\theta))$. L'objet intersecte un nombre paire de fois la demi-droite $[H, \vec{u}(\theta))$. Soient $\rho_1(\theta)$ et $\rho_2(\theta)$, les fonctions polaires correspondant aux deux premières intersections. $\rho_0(\theta)$ décrit la surface de $\mathcal{S}$.
Soient $\Delta v_0(\theta)$, $\Delta v_1(\theta)$ et $\Delta v_2(\theta)$, les aires des secteurs angulaires centrés en H et définis respectivement par $(\rho_0(\theta), d\theta)$, $(\rho_1(\theta), d\theta)$ et $(\rho_2(\theta), d\theta)$.

Le volume $\Delta v(\theta)$ correspondant à l'intersection de l'objet à déformer et du secteur angulaire centré en H, de direction $\vec{u}(\theta)$ et d'angle $d\theta$ vaut :

$$\Delta v(\theta) = \Delta v_2(\theta) - \Delta v_1(\theta)$$

On applique le même principe qu'en a) pour calculer les variations de volume de $\Delta v_1(\theta)$ et $\Delta v_2(\theta)$ dûes à la déformation :

$$\Delta v_1'(\theta) = \Delta v_0(\theta) + \Delta v_1(\theta)$$

$$\Delta v_2'(\theta) = \Delta v_0(\theta) + \Delta v_2(\theta)$$

Or, $\Delta v'(\theta)$ (volume correspondant à $\Delta v(\theta)$ après déformation) vaut :

$$\Delta v'(\theta) = \Delta v_2'(\theta) - \Delta v_1'(\theta) = \Delta v_2(\theta) - \Delta v_1(\theta)$$

Donc,

$$\Delta v'(\theta) = \Delta v(\theta)$$

Ainsi, il n'y a pas de variation de volume dans les sections de l'objet n'incluant pas H.

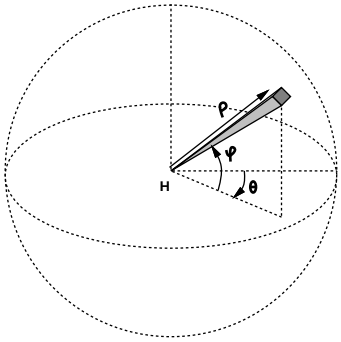

FIG. 3.6 – *Coordonnées sphériques utilisées pour le calcul de* Δv, Δv_0 *et* $\Delta v'$

L'extension à l'espace 3D est immédiate : en résolvant $\Delta v' = \Delta v_0 + \Delta v$ où $\Delta v'$, Δv_0, Δv sont les volumes de des secteurs angulaires 3D définis en coordonnées sphériques (figure 3.6) par $(\rho'(\theta,\varphi), d\theta, d\varphi)$, $(\rho_0(\theta,\varphi), d\theta, d\varphi)$ et $(\rho(\theta,\varphi), d\theta, d\varphi)$

$$\Delta\ v'(\theta,\varphi) = \frac{1}{3}\rho'(\theta,\varphi)^3 \cos\varphi d\theta d\varphi$$

$$\Delta\ v_0(\theta,\varphi) = \frac{1}{3}\rho_0(\theta,\varphi)^3 \cos\varphi d\theta d\varphi$$

$$\Delta\ v(\theta,\varphi) = \frac{1}{3}\rho(\theta,\varphi)^3 \cos\varphi d\theta d\varphi$$

nous obtenons :

$$\rho' = \sqrt[3]{{\rho_0}^3 + \rho^3}\ .$$

Ainsi, en utilisant cette équation, nous construisons une transformation qui conserve le volume le long d'un secteur angulaire. Cela implique que le volume de l'objet transformé est inchangé ou augmenté par le volume de $\mathcal{S}$.

- Contrôle de la déformation

L'utilisateur peut contrôler la déformation en changeant les paramètres qui définissent la forme de l'outil. Ainsi, il peut changer sa taille, son orientation et sa

position interactivement. Les figures 3.10 et 3.11 illustrent ces manipulations. La figure 3.11 met en évidence une autre particularité de la déformation par fusion : la topologie de l'objet est inchangée par la déformation (le tore déformé garde toujours sa topologie torique).

En appliquant successivement plusieurs déformations à un objet simple, des objets plus complexes peuvent être créés : la figure 3.12 montre un objet simple (une ellipsoïde) déformée par plusieurs outils pour modeler un chat (voir aussi la figure 3.13).

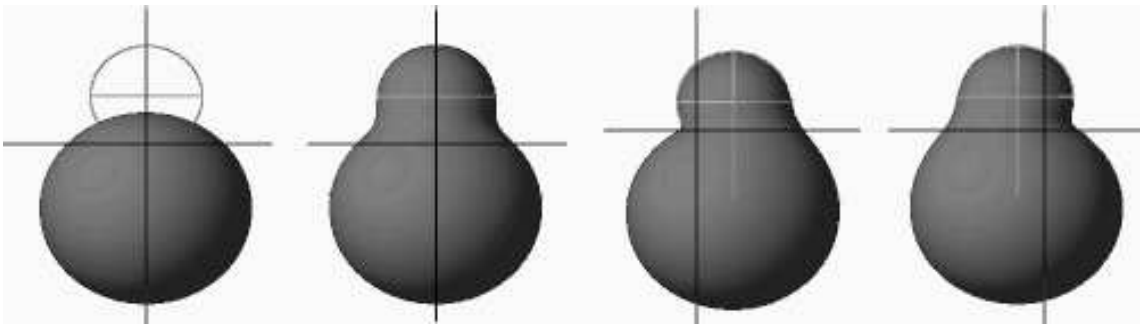

FIG. 3.7 – *Influence de la position du centre de l'outil. L'outil garde la même position, seule le centre de fusion bouge (croix noire)*

La position relative du centre de l'outil à l'intérieur de l'outil contrôle aussi la déformation. Par exemple, sur la figure 3.7, une sphère est déformée par un outil sphérique ; si le centre de l'outil est à l'intérieur de la sphère, elle est déformée de façon à inclure l'outil. Le volume de l'outil vient s'ajouter au volume de l'objet, et la position relative du centre de l'outil à l'intérieur de l'outil contrôle la direction principale dans laquelle le volume est ajouté.

3.1.3 Cas des objets décrits par un maillage

La déformation ainsi définie est théoriquement indépendante du type de représentation choisie pour l'objet à déformer. Nous allons nous intéresser plus particulièrement aux représentations polygonales, souvent utilisées en modélisation. Les objets sont représentés par un maillage de leur surface, c'est-à-dire des ensembles sommets/arêtes/facettes qui approximent leurs surfaces.

Afin de déformer un tel objet, chaque sommet est déplacé en lui appliquant la transformation (3.1) définie dans la section 3.1.2.1. Mais, le maillage résultant à souvent besoin d'être raffiné (i.e. ses facettes sont subdivisées, créant ainsi de

nouveaux sommets et de nouvelles arêtes, cf. figure 3.8) afin d'approximer de façon plus exacte la surface déformée. En effet, si on considère un cube, il peut être représenté par 6 facettes, 12 arêtes et 8 sommets ; on le déforme ensuite à l'aide d'un outil sphérique ; il faudra alors plus de 6 facettes pour approximer sa surface, en particulier au niveau de la zone déformée. Voir figure 3.9.

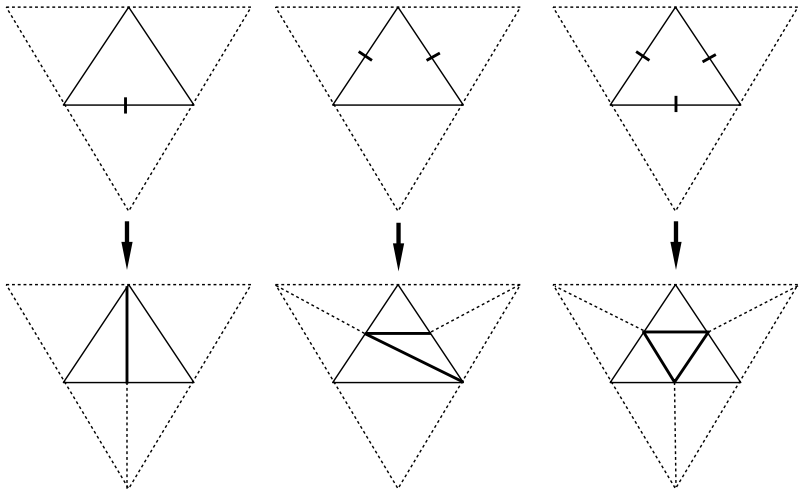

FIG. 3.8 – *Méthode de subdivision des facettes triangulaires : la facette est subdivisée en fonction des arêtes à subdiviser, les facettes adjacentes (en pointillés) sont subdivisées en conséquence*

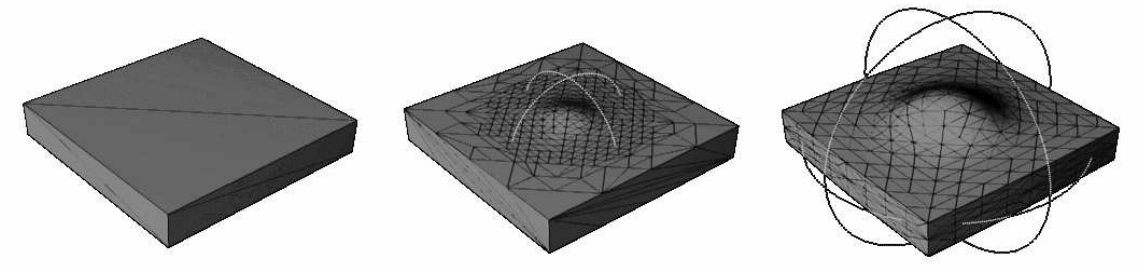

FIG. 3.9 – *Une boîte déformée par un outil sphérique, et son maillage raffiné en fonction de la déformation*

J. Griessmair et W. Purgathofer ont proposé une technique [GP89] permettant de raffiner le maillage utilisé pour représenter un objet déformé par une déformation B-spline à trois variables (déformation de type FFD). Cette technique se veut générale et pourrait être appliquée à toute déformation de l'espace. Mais l'algorithme

est complexe (en particulier le critère de raffinement) et coûteux en calcul : chaque itération de l'algorithme raffine uniquement deux facettes, il faut souvent de nombreuses itérations pour obtenir un maillage correspondant au niveau de raffinement souhaité. De plus, le critère de subdivision est basé sur une heuristique et l'erreur commise en approximant la surface idéale par ce maillage n'est pas calculée.

Nous ne souhaitons pas utiliser "le meilleur" algorithme de raffinement, mais plutot une méthode simple qui raffine le maillage de la surface déformée en un temps raisonnable. Nous avons donc développé une méthode simple, efficace et adaptée à la technique de déformation présentée ici.

Elle utilise le fait que la déformation est locale. Si $\rho \gg \rho_0$, la déformation est négligeable (voir section 3.1.2.2). Une zone d'influence de l'outil est définie par le critère $\frac{\rho}{\rho_0} < \alpha$ où α est spécifié par l'utilisateur ($\alpha > 1$). Si une facette est contenue ou intersecte la zone d'influence, elle est marquée et devra être subdivisée pour mieux coller à la déformation.

L'algorithme de subdivision est itératif. Nous connaissons la forme de l'outil, on peut donc établir un schéma de subdivision qui permettrait de coller au mieux à sa surface : un critère utile est la longueur des arêtes des facettes. A titre d'exemple, si la forme de l'outil est une sphère de rayon r, on peut considérer qu'un maillage qui approximera correctement sa surface devra avoir des arêtes dont la longueur est inférieure à βr, où β est aussi spécifié par l'utilisateur ($0 < \beta < 1$) en fonction de la finesse de l'approximation qu'il souhaite obtenir. Afin de raffiner le maillage de l'objet déformé, les facettes marquées sont subdivisées si au moins une de ses arêtes ne satisfait pas le critère de longueur. Ainsi, de nouvelles facettes et de nouveaux sommets sont créés ; ces sommets sont placés sur la surface déformée en utilisant la transformation (3.1). Cette opération est itérée jusqu'à ce que toutes les facettes marquées, ou nouvellement créées, satisfassent le critère.

Des résultats de cet algorithme sont montrés sur les figures 3.9 et 3.12. L'utilisateur peut ajuster la finesse du maillage en modifiant les paramètres α et β. Ce raffinement de maillage est aussi calculé en temps interactif[2], ce qui permet de juger directement de l'effet des paramètres sur le maillage.

2. Cette affirmation est à moduler en fonction de la puissance de la machine utilisée et du nombre de facettes à considérer, mais si on se contente d'appliquer l'algorithme aux facettes situées dans la zone soumise à l'influence de la déformation, on obtient de bonnes performances.

3.1.4 Intérêts

Un objet complexe peut être modelé interactivement et relativement rapidement en le déformant successivement un certain nombre de fois avec cet outil de déformation. Le contrôle de la déformation est intuitif pour l'utilisateur car il dépend directement de la forme de l'outil et de la position du centre de l'outil, paramètres que l'utilisateur peut modifier interactivement.

Les exemples présentés ici utilisent des outils sphériques et ellipsoïdaux. D'autres types d'outils de forme convexe, comme les cubes et le cônes, produisent des effets intéressants car ils créent des arêtes vives et des points coniques sur l'objet déformé (une discontinuité apparaît dans l'espace 3D déformé).

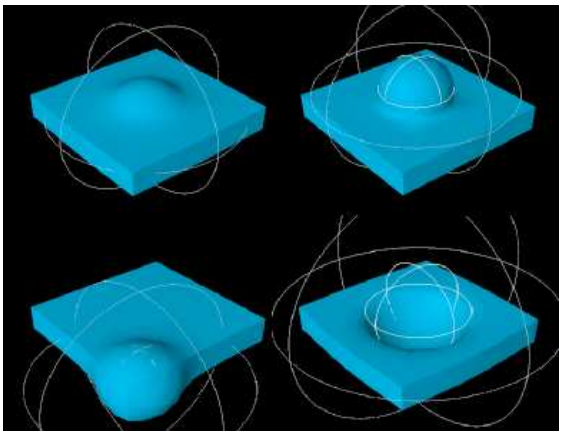

FIG. 3.10 – *Une boîte déformée par un outil sphérique*

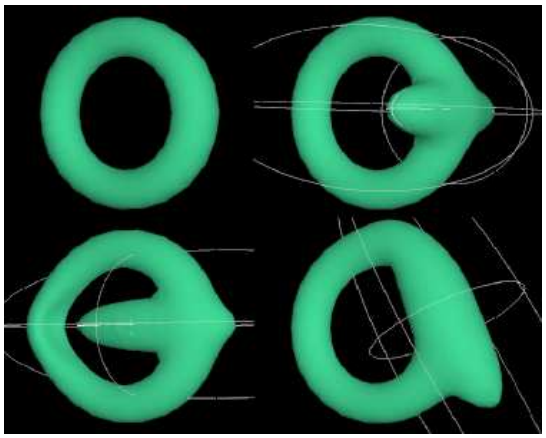

FIG. 3.11 – *Un tore déformé par un outil ellipsoïdal*

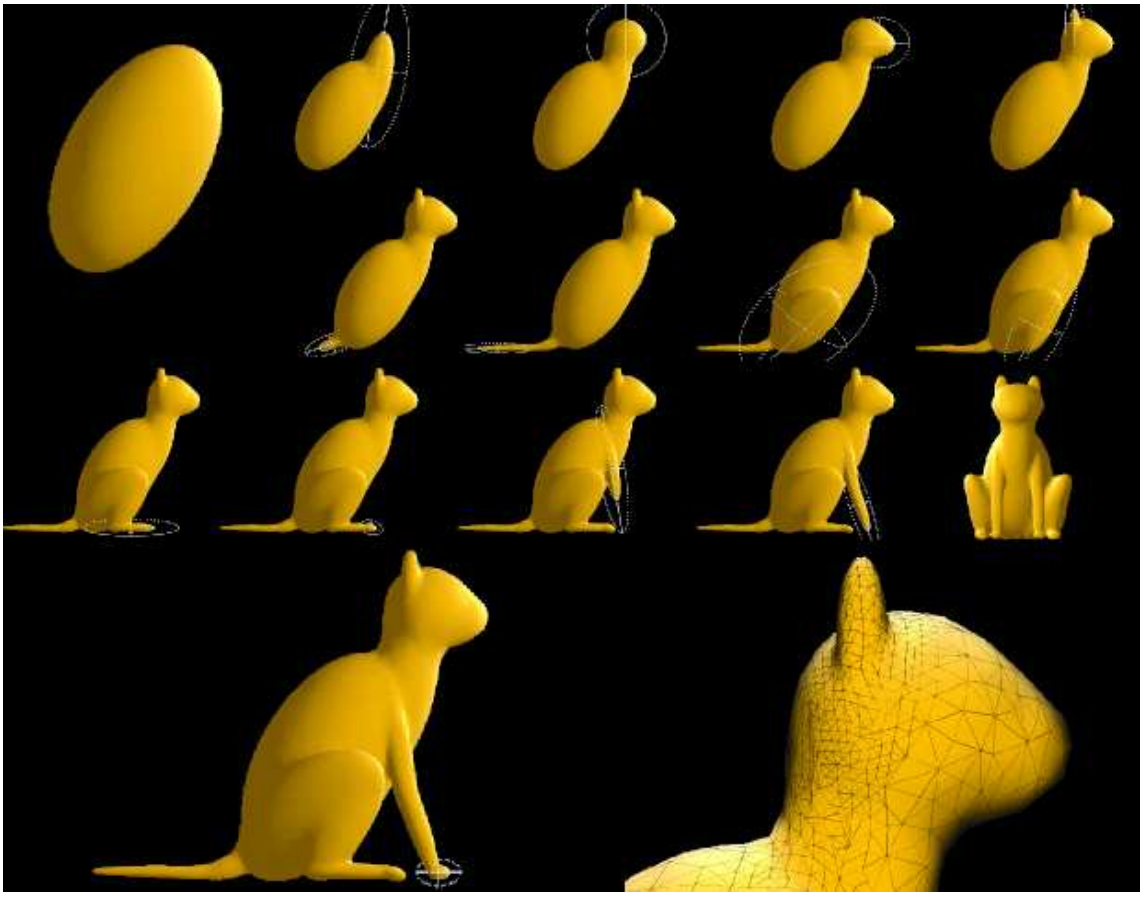

FIG. 3.12 – *Une ellipsoïde déformée en chat*

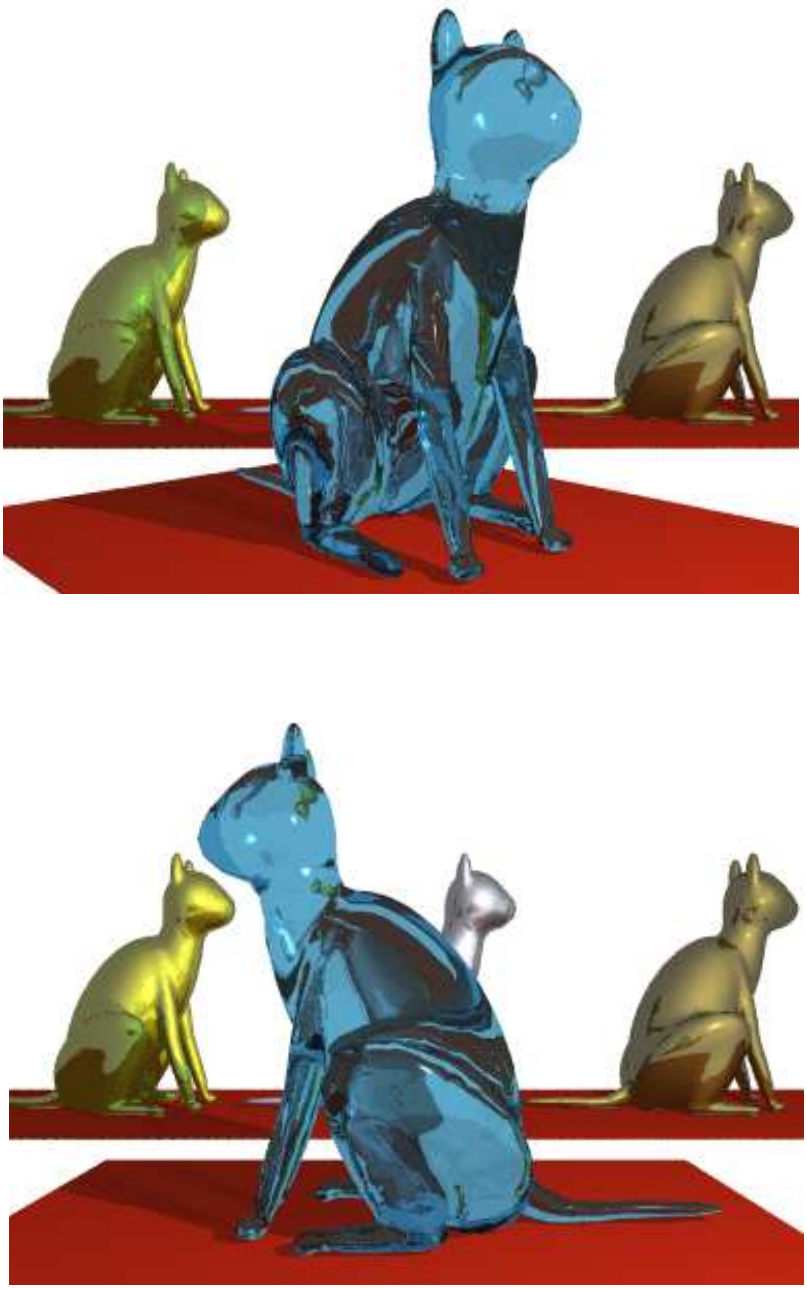

FIG. 3.13 – *Rendu "lancer de rayon" du chat*

3.2 Déformation par flexion

3.2.1 Aperçu

La flexion est une rotation par rapport à un axe dont l'effet est limité dans l'espace. L'idée est de pouvoir tourner (tordre) une partie d'un objet sans toucher au reste de l'objet. La transition de la zone subissant la flexion à la zone contenant le reste de l'objet est progressive.

Il s'agit d'une déformation d'espace qui sera utile par la suite pour gérer les articulations de nos modèles de type corps humains ou d'animaux.

Tout comme l'outil de fusion, le calcul de la déformation est simple et peu coûteux en temps, ce qui permet de manipuler l'outil de flexion interactivement.

Cet outil peut être comparé aux outils de courbure et de torsion proposés par Barr [Bar84]. L'effet obtenu par notre outil de flexion n'est pas le même que celui obtenu par l'outil de courbure de Barr. Si on applique notre outil de flexion à un tube, on provoque une cassure nette au niveau de l'axe central du tube (figure 3.15), alors qu'avec l'outil de courbure de Barr, on obtient une courbure constante de l'axe dans la zone courbée. Par contre, sur le plan du formalisme, notre outil de flexion se rapproche plus de la torsion de Barr. Pour Barr, la torsion s'exprime par $\theta' = f(z)$, et, dans notre cas, elle s'exprime par $\theta' = f(\theta)$ (cf. section suivante).

3.2.2 La technique de déformation par flexion

3.2.2.1 Description

Tout comme la déformation par fusion, la flexion est une déformation de l'espace. Elle est définie par un repère orthonormé $(O, \vec{\imath}, \vec{\jmath}, \vec{k})$, un angle α qui permettra de définir une zone d'influence ($\alpha \in \ ]0, \pi]$) et l'angle de flexion φ tel que $-\alpha < \varphi < \alpha$ (figure 3.14).

La flexion s'effectue par rapport au repère $(O, \vec{\imath}, \vec{\jmath}, \vec{k})$:
O est le centre de la flexion (correspondant au point d'articulation) ,
$\vec{k}$ défini l'axe par rapport auquel la flexion va s'effectuer,
$\vec{\imath}$ défini la direction principale qui va subir la flexion d'angle φ de telle sorte que tout point M appartenant au plan $(O, \vec{\imath}, \vec{k})$ subira une rotation par rapport à $\vec{k}$ d'angle exactement égal à φ si $\overrightarrow{OM}$ est dans la direction de $\vec{\imath}$ ($\overrightarrow{OM} \cdot \vec{\imath} > 0$), et ne subira pas de rotation sinon ($\overrightarrow{OM} \cdot \vec{\imath} \leq 0$).

La flexion est restreinte à une zone contenant la demi-droite $[O, \vec{\imath})$, et est délimitée par deux demi-plans $(O, \vec{e_1}, \vec{k})$ et $(O, \vec{e_2}, \vec{k})$, où $\vec{e_1}$ (resp. $\vec{e_2}$) est obtenu

en appliquant au vecteur $\vec{\imath}$ une rotation d'angle α (resp. $-\alpha$) par rapport à $\vec{k}$ (figure 3.14).

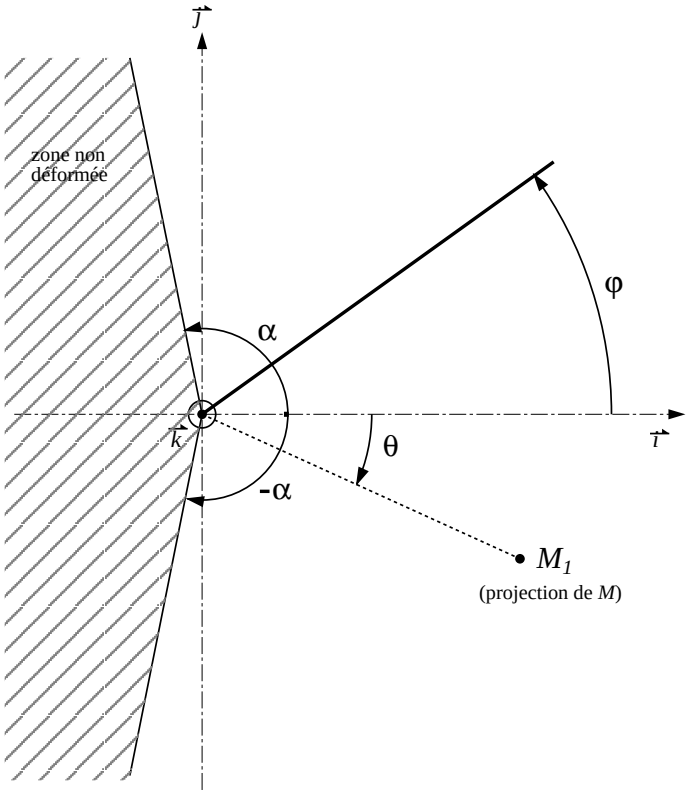

FIG. 3.14 – *Notations utilisées pour définir la flexion*

En coordonnées sphérique, un point M de l'espace est repéré par ses coordonnées (ρ, θ, ψ) avec $\rho = \|\overrightarrow{OM}\|$, $\theta = (\vec{\imath}, \widehat{\overrightarrow{OM_1}})$ et $\psi = (\widehat{\overrightarrow{OM_1}, \overrightarrow{OM}})$, où M_1 est la projection orthogonale de M sur le plan $(O, \vec{\imath}, \vec{\jmath})$.
θ est compris entre $-\pi$ et π, et ψ est compris entre $-\frac{\pi}{2}$ et $\frac{\pi}{2}$.

La flexion transforme le point M en un point M' de coordonnées (ρ, θ', ψ). Les coordonnées ρ et ψ sont inchangées, seule la coordonnée θ' varie (en fonction du paramètre θ uniquement) :

$$\theta' = f(\theta) \tag{3.2}$$

où f définie sur $]-\pi, \pi]$ est telle que,

- si $\theta < -\alpha$ ou $\theta > \alpha$ alors $f(\theta) = \theta$; la flexion ne modifie pas les points situés en dehors de la zone d'influence ;

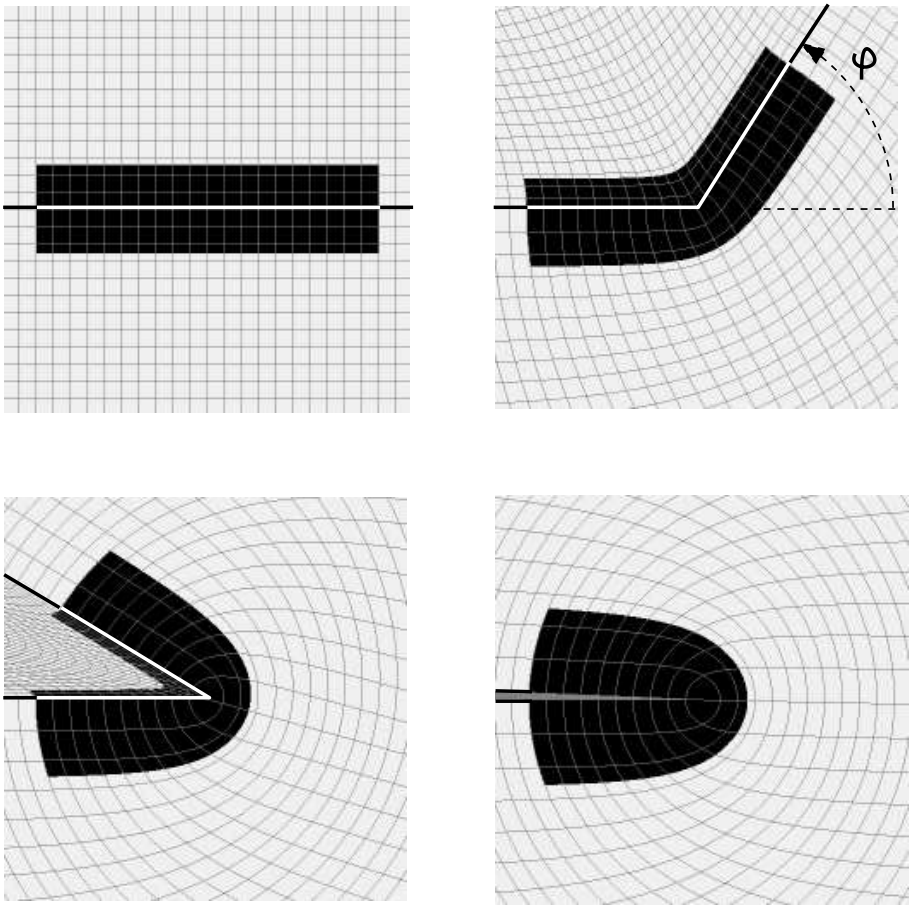

FIG. 3.15 – *Effet de la flexion sur un objet rectangulaire ; vue en coupe de la déformation de l'espace dans un plan parallèle au plan* $(O, \vec{\imath}, \vec{\jmath})$ *; déformations obtenues pour différentes valeurs de* φ *et avec* $\alpha = \pi$

– si $-\alpha \leq \theta \leq \alpha$ alors $f(\theta)$ varie continûment de $-\alpha$ à α de telle sorte que $f(0) = \varphi$ et que f soit strictement croissante- cette fonction permet de définir une rotation dont l'angle varie en fonction de la position de M à l'intérieur de la zone d'influence de telle sorte qu'il vaut zéro sur les bords et φ quand M se trouve dans le plan médian. f doit être strictement croissante pour que la transformation soit bijective, on s'assure ainsi qu'un objet qui subit cette transformation ne peut pas se *replier* sur lui-même (voir figure 3.18). La fonction f linéaire par morceau de la figure 3.16 est une solution acceptable, mais on lui préférera une fonction $\mathcal{C}^1$ telle que celle de la figure 3.17, elle permet de préserver la continuité lors de la transformation (cf. paragraphe suivant).

La figure 3.15 montre comment agit la transformation dans un plan parallèle

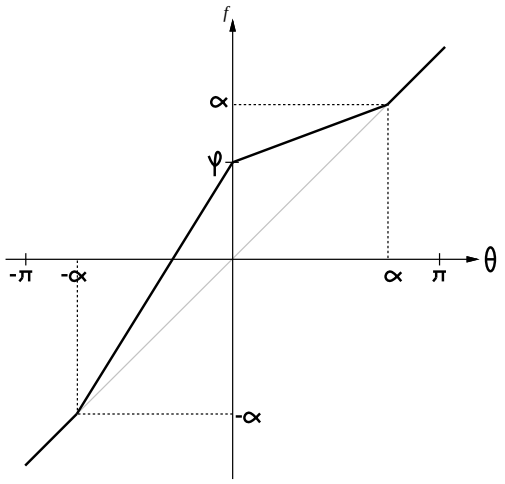

FIG. 3.16 – *Fonction $f(\theta)$ linéaire par morceaux*

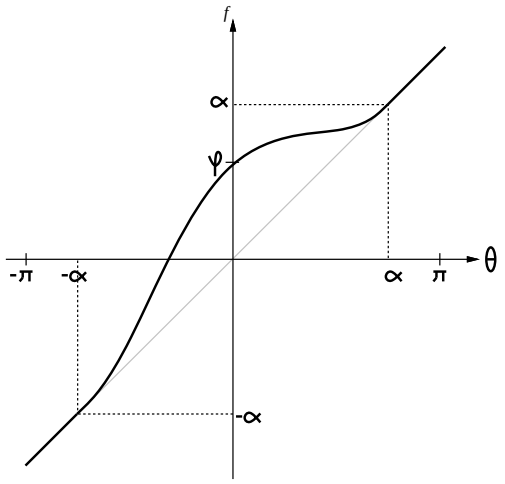

FIG. 3.17 – *Fonction $f(\theta)$ de continuité $\mathcal{C}^1$*

au plan $(O, \vec{i}, \vec{j})$. Nous verrons, dans le chapitre suivant, comment elle est utilisée pour simuler les articulations dans le cadre de formes organiques articulées.

3.2.2.2 Propriétés

- Continuité

Cette transformation va servir à déformer un objet en déformant continûment l'espace dans lequel il est placé. En utilisant une fonction $f(\theta)$ de continuité $\mathcal{C}^1$ pour tout θ, on s'assure que si la surface de l'objet est continûment différentiable, alors la surface de l'objet déformé est continûment différentiable au moins à l'ordre 1. Cette

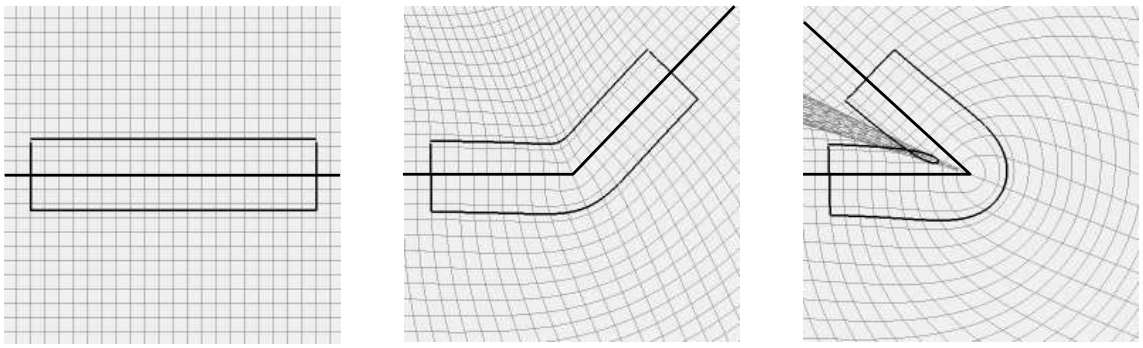

FIG. 3.18 – *Conséquence de l'utilisation d'une fonction f non strictement croissante : l'objet rectangulaire s'auto-intersecte*

propriété est importante pour conserver un aspect lisse aux objets qui avaient cet aspect avant la déformation.

- Localité

La transformation a été construite de telle sorte que seuls les points inclus dans un secteur angulaire délimité par deux plans. On remarque quand même qu'un point situé à l'infini sur la demi-droite $[O, \vec{i})$ subira la transformation. Cette "localité" est donc toute relative.

Il peut s'avérer intéressant, en particulier dans le cadre des outils de déformation, de limiter aussi la déformation à l'intérieur d'une zone fermée incluant l'articulation. Nous proposerons une telle extension dans le section 3.3.

- Conservation du volume

Rien ne garanti que le volume d'un objet qui subit une flexion soit conservé. Nous avons cherché à obtenir cette propriété en modifiant à la fois θ et ρ pendant la transformation de telle sorte que le volume inclus dans un secteur angulaire soit conservé, mais les résultats n'ont pas été concluants, la flexion ne ressemblait plus à une flexion.

3.2.3 Cas des objets décrits par un maillage

Comme pour les déformations par fusion, si l'objet à déformer est défini par un maillage polyédrique de sa surface, il peut être intéressant de raffiner le maillage pour approximer au mieux la surface déformée.

L'algorithme s'inspire de celui décrit en 3.1.3, mais le critère de subdivision change. Il est basé sur la courbure locale : une arête $[A, B]$ est subdivisée si $\|\vec{n_B} - \vec{n_A}\| < \varepsilon$ où $\vec{n_A}$ et $\vec{n_B}$ sont les normales associées à A et à B, et où ε est un paramètre fixé par l'utilisateur et permet de raffiner plus ou moins le maillage.

3.2.4 Intérêt

Outre le type de déformation qu'il engendre, l'outil de flexion va nous permettre de résoudre le problème lié au glissement des points à la surface d'un objet au niveau d'une zone articulée. Ce problème se remarque en particulier lorsqu'une texture a été plaquée sur l'objet, celle-ci semble glisser sur la surface lorsqu'on anime l'articulation. Nous allons illustrer nos propos par un exemple.

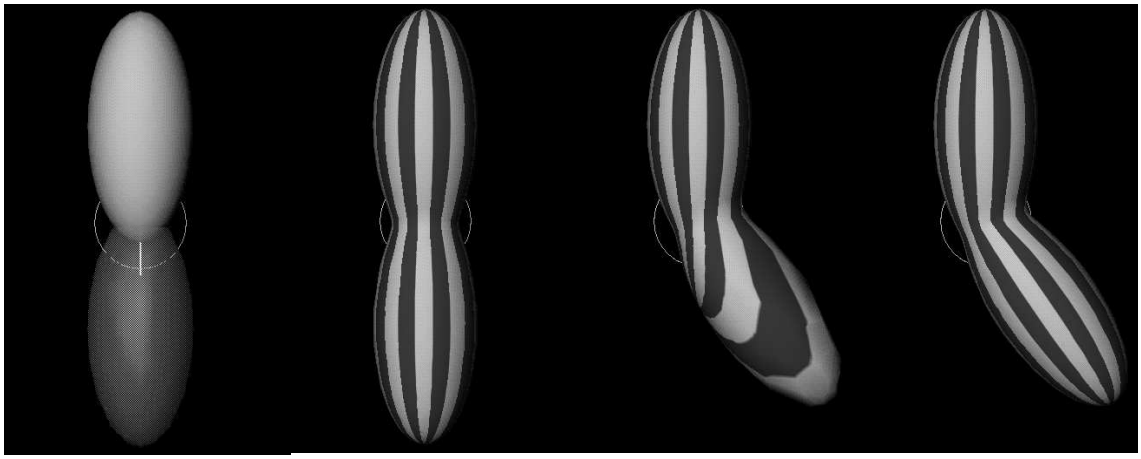

FIG. 3.19 – *Problème du glissement des textures : a) & b) une articulation est crée en fusionnant deux ellipsoïdes ; l'articulation est animée - c) en tournant l'outil de fusion uniquement - d) en utilisant une flexion*

Une articulation simple entre deux ellipsoïdes est créée en appliquant une déformation de type fusion à un objet ellipsoïdal, l'outil de fusion a lui aussi une forme ellipsoïdale et le centre de fusion se trouve au niveau du point d'articulation (figure 3.19-a). On se place dans le cas où l'objet est représenté par un maillage de sa surface, ce maillage est raffiné pour coller au mieux à l'objet déformé. Une texture est alors plaquée sur l'objet ainsi créé (à chaque sommet du maillage est associé une coordonnée texture, (figure 3.19-b). Pour animer l'articulation, on pourrait penser qu'il suffit d'appliquer une rotation à l'outil de fusion et de recalculer la déformation, mais on s'apperçoit alors que la texture semble glisser sur la surface (figure 3.19-c). En fait, c'est le maillage sous-jacent qui "glisse". Si, par contre, au lieu d'appliquer une rotation à l'outil de fusion, on applique une déformation de type flexion au niveau de l'articulation, alors, lorsqu'on modifie l'angle de la flexion, on obtient l'effet d'articulation et la texture reste "collée" à la partie qui pivote (figure 3.19-d). Ceci est dû au fait que tout l'espace dans lequel est placé l'objet se tord, le maillage suit et par conséquent la texture aussi.

3.3 Extension : ajout d'une zone d'influence

Pour un outil de déformation, il est intéressant que l'utilisateur puisse définir une zone d'influence, de sorte que seuls les points situés à l'intérieur de cette zone subiront la déformation. Il est, de plus, souhaitable qu'il y ait une zone la transition entre la zone où la déformation agira pleinement et celle où elle n'agira pas du tout. Nous allons proposer une méthode simple permettant de définir une zone d'influence pour l'outil de fusion et l'outil de flexion.

Pour cela, l'utilisateur spécifie deux zones convexes[3] Z_1 et Z_2 imbriquées l'une dans l'autre ($Z_1 \subset Z_2$) et contenant le centre de fusion ou le centre de flexion (en fonction de la déformation considérée) ; ce point sera noté C dans la suite. On peut prendre, par exemple, l'intérieur de deux ellipsoïdes dont l'une est une homothétie de l'autre par rapport à C (c'est le cas que nous avons choisi d'implémenter). Ces zones vont nous servir à définir une fonction d'influence

$$h : \mathbb{R}^3 \longrightarrow \mathbb{R}^+$$

qui vaudra 1 à l'intérieur de Z_1, 0 à l'extérieur de Z_2 et passera continûment de 1 à 0 entre Z_1 et Z_2.

Pour un point M de l'espace,
notons I_1 (resp. I_2) l'intersection de la demi-droite $[CM)$ avec la frontière de Z_1 (resp. Z_2),
notons $R_1 = \|CI_1\|$ et $R_2 = \|CI_2\|$,
notons $\rho = \|CM\|$,
alors,

$$h(M) = \begin{cases} 1 \text{ si } \rho \leq R_1 \\ 0 \text{ si } \rho \geq R_2 \\ 2\left(\frac{\rho - R_1}{R_2 - R_1}\right)^3 - 3\left(\frac{\rho - R_1}{R_2 - R_1}\right)^2 + 1 \quad \text{si} \quad R_1 < \rho < R_2 \end{cases}$$

On peut alors prendre en compte le zone d'influence dans la transformation (3.1) utilisée pour la déformation par fusion de la façon suivante :

$$\rho' = \sqrt[3]{\rho^3 + (h(M).\rho_0)^3} \tag{3.3}$$

De même, on peut prendre en compte le zone d'influence dans la transformation (3.2) utilisée pour la déformation par flexion de la façon suivante :

$$\theta' = h(M).f(\theta) \tag{3.4}$$

3. l'extension aux zones étoilées ne pose pas de problème.

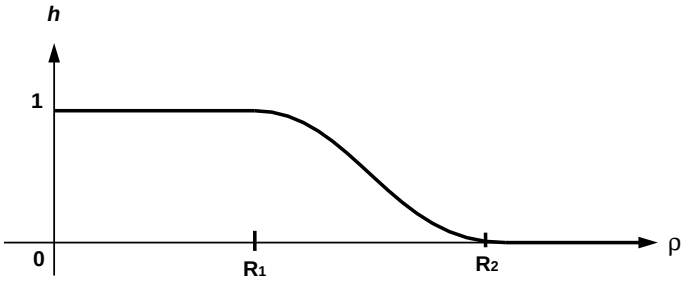

FIG. 3.20 – *Variations de la fonction d'influence h pour un point M pris le long d'une demi-droite partant de C et de direction fixée*

Il est à noter que, dans le cas de la fusion, la propriété de conservation du volume n'est plus garantie quand on prend en compte la zone d'influence.

Les figures 3.22 et 3.21 montre l'effet de la zone d'influence sur la flexion et la fusion.

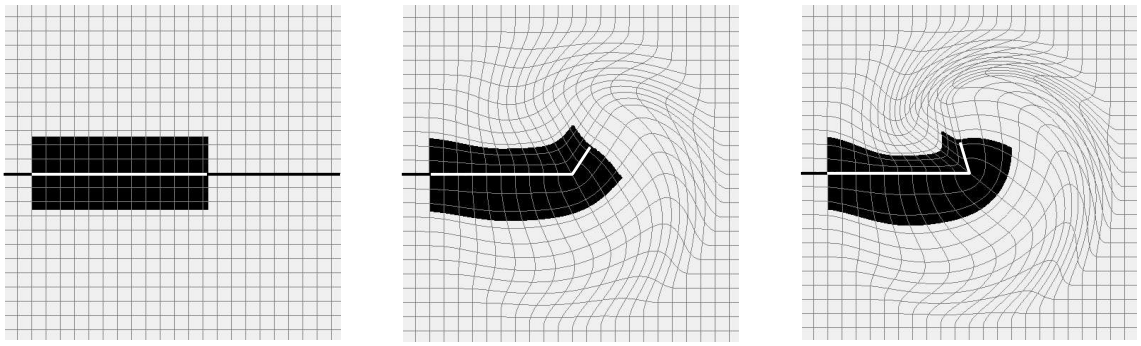

FIG. 3.21 – *Flexion : effet d'une zone d'influence sphérique centrée au point d'articulation (vue en coupe)*

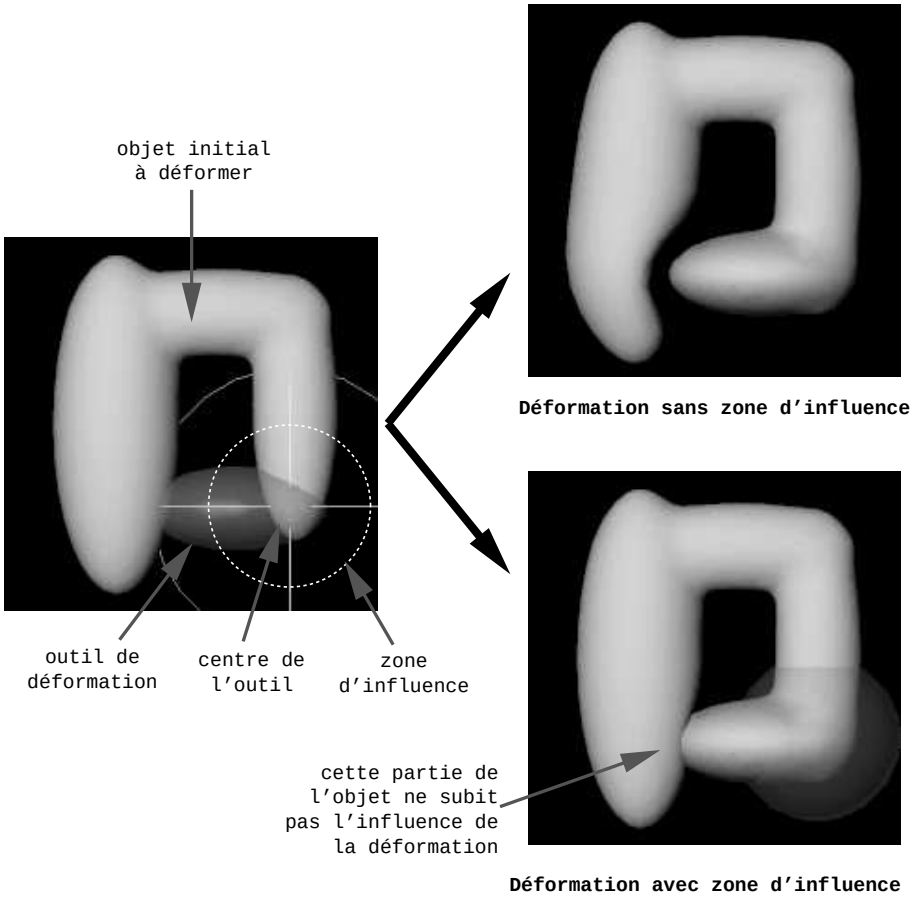

FIG. 3.22 – *Fusion : effet d'une zone d'influence sphérique*

Chapitre 4

Modèle dédié de type fusion/flexion

4.1 Aperçu

Les outils de déformation (fusion et flexion) présentés dans le chapitre précédent permettent de modeler facilement des formes organiques telles que les corps humains ou d'animaux (cf. l'exemple du chat, figure 3.12). Nous allons voir comment utiliser ces techniques pour définir un modèle dédié à ce type d'objets 3D apportant à l'utilisateur la souplesse nécessaire pour modeler, animer et texturer les objets ainsi créés.

Le principal intérêt de nos outils de fusion et de flexion est d'offrir à l'utilisateur un contrôle de haut niveau sur les déformations : il agit sur la forme fusionnée ou sur l'angle de la flexion, et non pas directement sur la surface de l'objet déformé. Les problèmes de continuité de la surface sont alors gérés automatiquement. Le second intérêt est qu'une représentation polyédrique de la surface de l'objet peut être obtenue directement.

Le modèle adapté aux formes organiques articulées que nous allons décrire, a les caractéristiques suivantes :

- le modèle est animable : en agissant sur les paramètres des déformations, le modèle s'anime en garantissant que le maillage et la texture de l'objet suivent correctement l'animation ;

- le modèle est multirésolution : il permet d'obtenir un maillage plus ou moins fin de la surface l'objet, globalement (affecte le maillage de tout l'objet) ou localement (une zone particulière du maillage est raffinée).

- la construction du modèle est interactive : le modéliste construit et affine interactivement son modèle et voit le résultat en temps interactif.

4.2 Définition du modèle

Le modèle est défini par la donnée d'une forme initiale ayant une représentation analytique simple (par exemple, une sphère) et d'une succession d'outils de déformation. La construction de l'objet se fait alors en appliquant successivement les diverses déformations à l'objet.

Par exemple, un modèle très simplifié de bras pourrait être défini par une forme initiale sphérique correspondant à l'épaule, de deux outils de fusion ellipsoïdaux correspondant aux deux parties du bras, de deux outils de flexion pour les articulations au niveau de l'épaule et du coude.

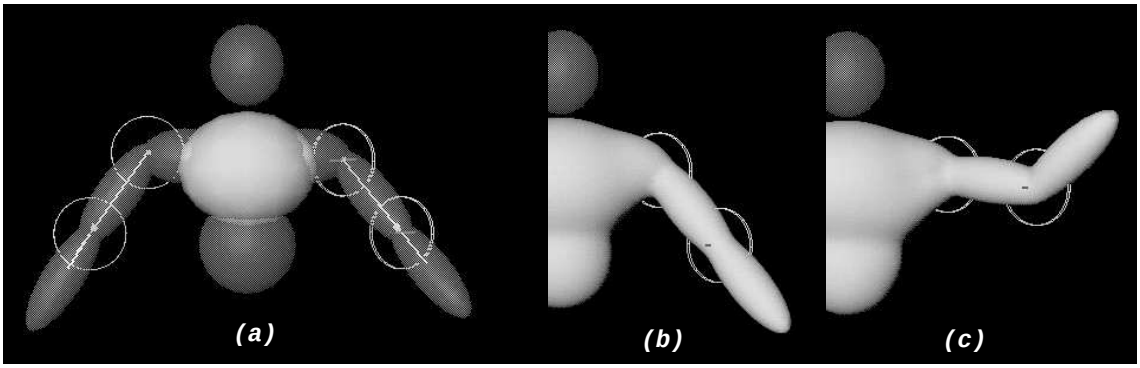

FIG. 4.1 – *Modèle de bras simplifié : a) la sphère initiale (gris clair) et les outils de fusion (ellipsoïdes gris foncées) et de flexion (cercles blancs) ; b) la forme obtenue ; c) après animation des articulations.*

Le maillage de la surface de l'objet est généré de façon à respecter les critères suivants :

1. le maillage ne glisse pas sur la surface de l'objet lorsque l'objet est animé ;

2. le maillage peut être généré simplement à n'importe quel niveau de détail.

Le premier critère assure qu'une texture appliquée sur l'objet ne glissera pas de façon anormale en cours d'animation.

Le second critère permet de créer interactivement un objet. Un maillage de sa surface peut-être obtenu directement. Un maillage grossier est utilisé pour la visualisation interactive du résultat pendant la phase de création, alors qu'un maillage plus fin sera utilisé pour le calcul du rendu final. L'utilisateur peut contrôler la finesse du maillage localement ou globalement.

Pour atteindre cet objectif, la forme initiale doit respecter les deux critères et les déformations successives doivent les préserver.

4.3 La forme initiale

La forme initiale est définie par deux fonctions v et $\vec{n}$, et par un maillage simple à facettes triangulaire[1] qui correspond au niveau de détail zéro. Les deux fonctions permettent d'ajouter de nouveaux sommets et leurs normales au maillage afin de le raffiner pour approximer la surface de la forme plus finement.

L'algorithme de subdivision utilisé pour raffiner le maillage coupe les arêtes des facettes triangulaires en deux (voir sections 4.5 et 3.1.3). Une arête peut être vue comme une approximation d'une courbe dessinée sur la surface exacte de la forme (figure 4.2. Un nouveau sommet et sa normale associée sont créés pour chaque arête subdivisée.

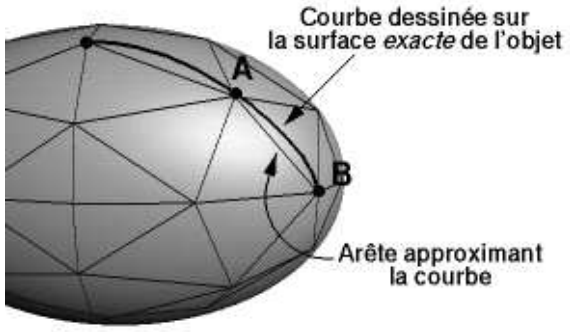

FIG. 4.2 – *Approximation d'une courbe dessinée sur la surface exacte d'un objet*

1. nous n'utiliserons que des maillages à facettes triangulaires, notre méthode ne nécessite pas de maillage à facettes plus complexes.

Soit A et B les deux extrémités d'une arête à subdiviser. La fonction $v(A, B)$ calcule le point 3D de la surface qui correspond à un point intermédiaire sur la courbe exacte approximée par l'arête. La fonction $\vec{n}(A, B)$ calcule la normale à la surface en ce point.

De telles fonctions peuvent être facilement construites pour certaines classes de formes initiales intéressantes dans notre cas. Nous allons en donner deux exemples.

4.3.1 Forme étoilée

Déterminons v et $\vec{n}$ pour une forme étoilée.

$v(A, B)$ = intersection entre la surface de la forme et la demi-droite $[CM)$ où C est le centre de l'étoile et $M = \dfrac{A+B}{2}$ (cette intersection est unique car la forme est étoilée).

$\vec{n}(A, B)$ = normale à la surface au point $v(A, B)$.

La plus simple (et aussi la plus utile) des formes initiales de ce type est la sphère (voir figure 4.3).

$v(A, B) = C + r\dfrac{\overrightarrow{CM}}{\|\overrightarrow{CM}\|}$ où $M = \dfrac{A+B}{2}$ et r est le rayon de la sphère.

$\vec{n}(A, B) = \dfrac{\overrightarrow{CM}}{\|\overrightarrow{CM}\|}$

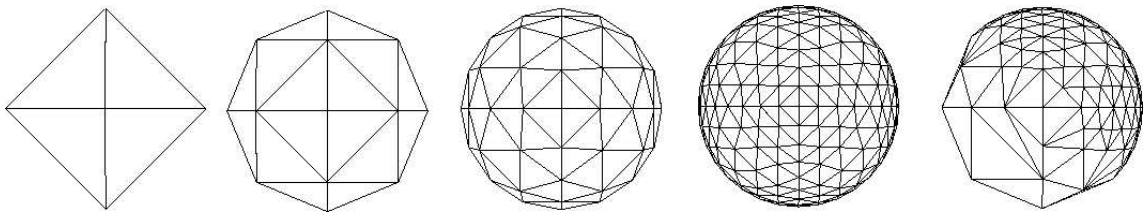

FIG. 4.3 – *Une sphère : à gauche, le maillage correspondant au niveau zéro, puis différents niveaux de raffinement ; à droite, seule la partie supérieure droite de la sphère a été raffinée (l'algorithme de subdivision est décrit en section 3.1.3)*

4.3.2 Surfaces paramétriques

La surface de la forme initiale est décrite par une fonction paramétrique à deux variables $f(s, t)$.

$v(A, B) = f(\dfrac{s_A + s_B}{2}, \dfrac{t_A + t_B}{2})$ où (s_A, t_A) et (s_B, t_B) sont les paramètres

correspondant respectivement à A et à B.

$\vec{n}(A,B) = \frac{\vec{g}(\frac{s_A+s_B}{2}, \frac{t_A+t_B}{2})}{\|\vec{g}(\frac{s_A+s_B}{2}, \frac{t_A+t_B}{2})\|}$ où $\vec{g}(s,t)$ est un vecteur normal à la surface au point $f(s,t)$ $\left(\vec{g}(s,t) = \frac{\partial f}{\partial s}(s,t) \wedge \frac{\partial f}{\partial t}(s,t) \right)$.

4.4 Déformations de la forme initiale

Afin modeler la forme initiale pour obtenir la forme souhaitée, on lui applique une succession de déformations. Chacune déforme l'espace 3D dans lequel repose la forme initiale. Les déformations que nous allons utiliser sont celles définies dans le chapitre précedent : la fusion et la flexion.

Soit M_0 un sommet du maillage de la surface de la forme initiale,
soient $(F_i)_{i=1,\cdots,k}$ les k fonctions correspondant aux déformations de l'espace appliquées successivement pour modeler l'objet,
le point M sur la surface de l'objet final correspondant à M_0 est donné par :

$$M = F_k \circ F_{k-1} \circ \cdots \circ F_1 (M_0)$$

Pour obtenir un nouveau point sur le maillage final ne correspondant pas directement à un sommet du maillage initial (niveau de détail supérieur à zéro), il suffit d'appliquer la même transformation à un point résultant de la fonction v d'ajout de nouveaux sommets :
soient A_0 et B_0 deux extrémités d'une arête du maillage initial,
le point M' sur la surface de l'objet final correspondant à un point intermédiaire sur l'arête $[A_0, B_0]$ est donné par :

$$M' = F_k \circ F_{k-1} \circ \cdots \circ F_1 (v(A_0, B_0))$$

La fonction $\vec{n}(A_0, B_0)$ peut-être utilisée pour calculer directement la normale exacte en M' .

4.5 Niveaux de détail

Différents niveaux de détail du maillage d'un même objet peuvent être obtenus directement à partir de la description mathématique de sa forme.

Le niveau de détail le plus grossier est donné par le maillage associé initialement à l'objet (cf. section 4.3). Le niveau de détail supérieur est calculé en subdivisant les facettes du maillage de manière similaire à celle exposée en 3.1.3, c'est-à-dire en fonction de critères.

Nous allons appliquer ici deux types de critères :

- Le premier s'appuie sur la longueur des arêtes et a déjà été évoqué en 3.1.3. Il contrôle la finesse du maillage dans les zones où les outils de fusion ont opérés.

  Le processus est le suivant : une arête est subdivisée si sa longueur est supérieure à β où β est fixé par l'utilisateur.

- Le second critère est basé sur la courbure locale et a déjà été évoqué en 3.2.3. Il est utilisé pour obtenir un meilleur lissage du maillage que ne le permet le premier critère seul. Il permet à l'utilisateur d'obtenir un maillage qui aussi proche qu'il le souhaite de la surface exacte de l'objet.

  Le processus est le suivant : une arête $[A, B]$ est subdivisée si $\|\vec{n_B} - \vec{n_A}\| < \varepsilon$ où $\vec{n_A}$ et $\vec{n_B}$ sont les normales associées à A et à B, et où ε est fixé par l'utlisateur.

Le processus de raffinement peut être effectué, soit de façon globale à tout l'objet, soit de façon plus locale. Dans ce dernier cas, l'utilisateur défini une zone d'intérêt à l'intérieur de laquelle le maillage est suceptible d'être raffiné en fonction des critères spécifiés.

La figure 4.4 montre un exemple de maillage généré à différents niveaux de détails.

4.6 Résultats

La figure 4.6 montre le type d'objet qu'il est possible de créer en utilisant notre modèle. Ce modèle ne peut pas décrire une classe de formes aussi étendue que celle générée par les modèles implicites. En particulier, les changements topologiques ne sont pas possibles. Malgré tout, notre modèle est bien adapté à la modélisation des formes dites organiques, et ce, de façon simple et intuitive pour l'utilisateur.

Afin de manipuler de tels modèles, l'interface du modeleur doit permettre à l'utilisateur d'agir interactivement sur les paramètres suivants : la forme, le centre et la zone d'influence de chaque outil de fusion ainsi que l'angle et la zone d'influence de chaque outil de flexion. Elle doit permettre de définir les zones où le maillage sera raffiné, ou définir un raffinement global du maillage. Elle doit aussi permettre de

définir comment la texture sera plaquée sur la surface de l'objet (figure 4.5). Enfin, elle doit permettre d'animer le modèle (figure 4.6).

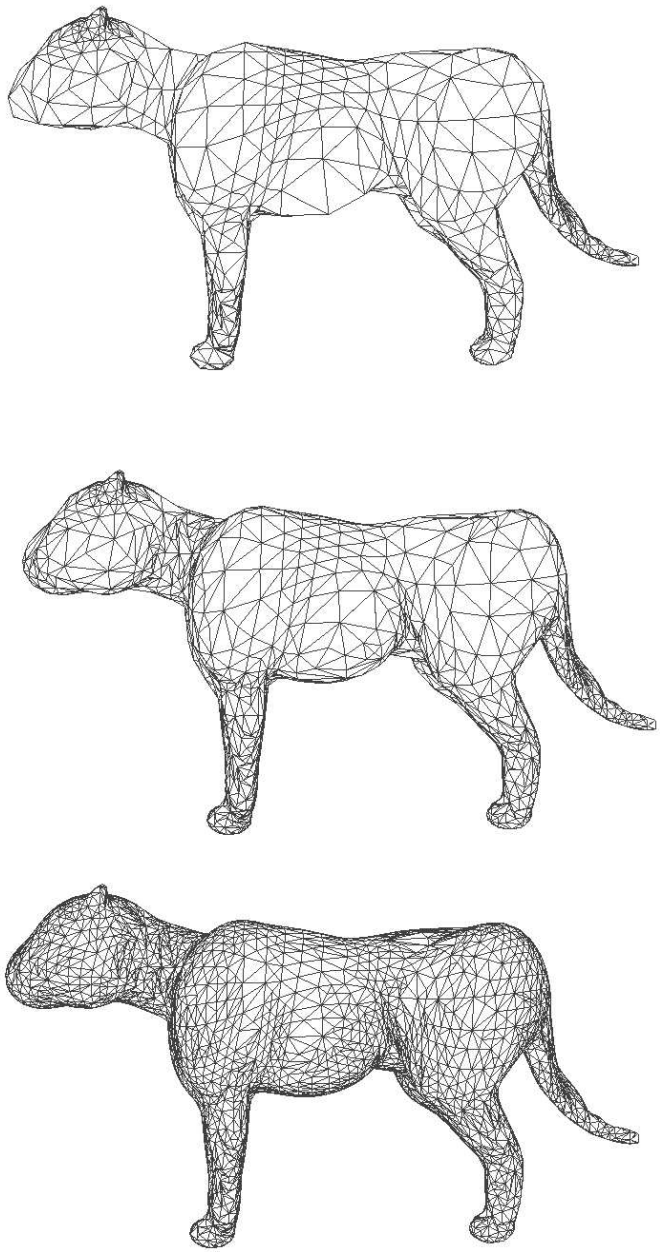

FIG. 4.4 – *Un modèle de tigre maillé à différents niveaux de détails (respectivement 2594, 4290 et 10866 facettes)*

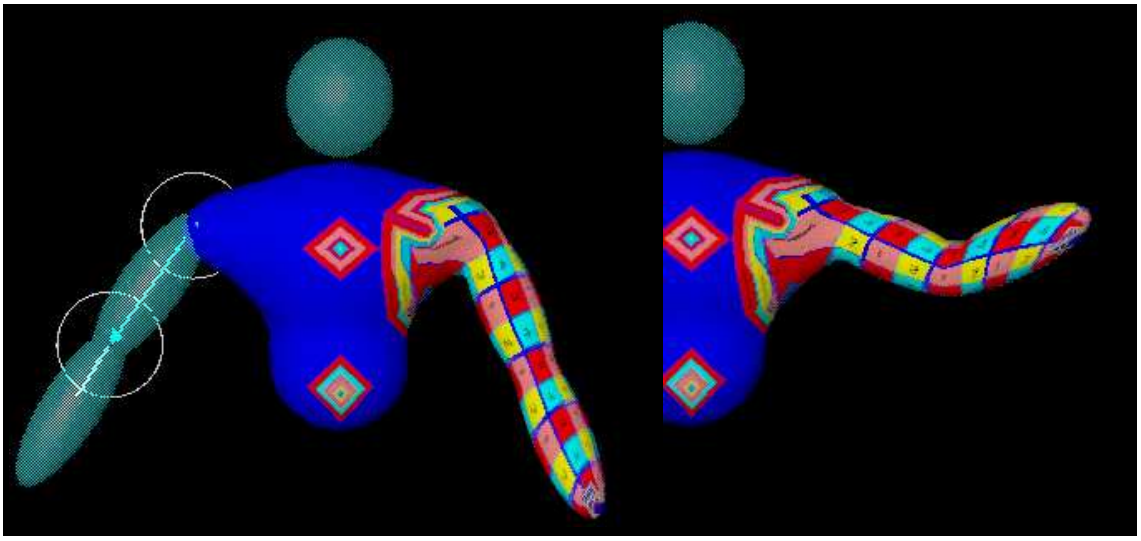

FIG. 4.5 – *Un bras articulé et texturé*

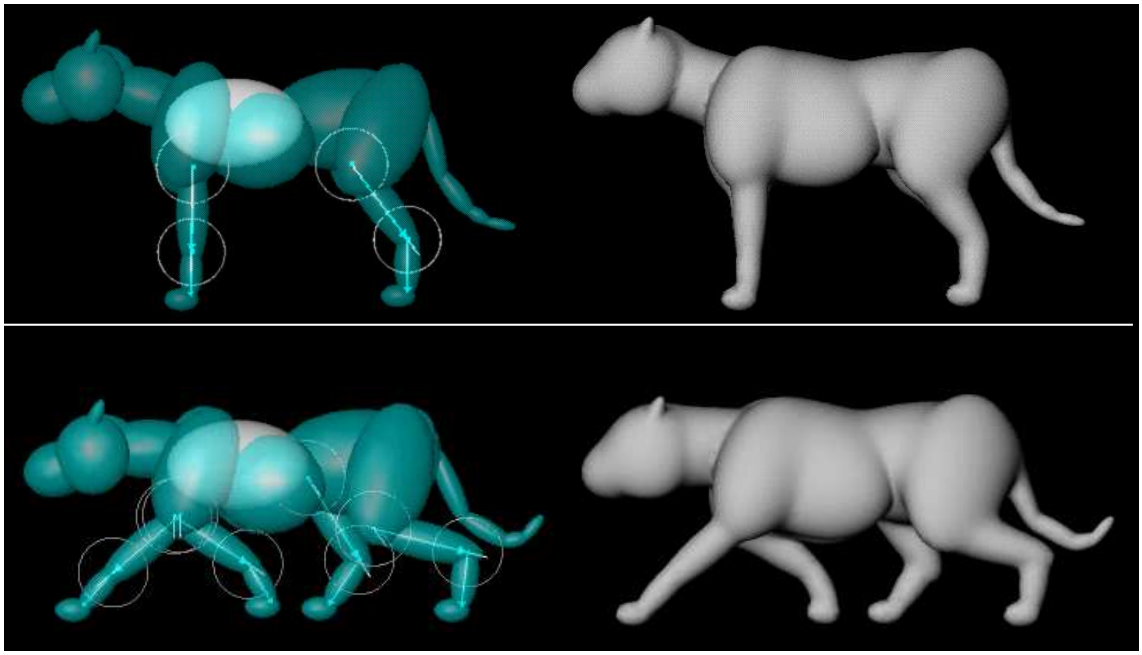

FIG. 4.6 – *Modèle de tigre articulé*

Chapitre 5

Outils interactifs et Implémentation

5.1 Objectif

Après avoir présenté les techniques de modélisation que nous proposons, nous allons décrire, dans ce chapitre, la façon dont nous avons implémenté les outils interactifs décrits dans les chapitres précédents.

La principale difficulté pour tester une méthode de modélisation interactive est justement de développer l'interface graphique qui va permettre à l'utilisateur de créer et modeler des objets 3D. Les interfaces permettant de manipuler des objets 3D figurent parmi les plus complexes à réaliser. Ceci est principalement dû au fait qu'il est difficile d'interagir avec une scène 3D à travers une interface essentiellement 2D : souris+clavier en entrées et écran en sortie.

Nous nous sommes fixés, comme objectif, de ne pas réinventer la roue et d'utiliser le plus possible les librairies et programmes informatiques existants. Ainsi, nous limitons les développements et profitons d'une base de composants logiciels conséquente et stable. En contrepartie, il faut apprendre à utiliser ces composants et se plier aux règles qu'ils imposent. On y perd donc en souplesse de programmation, mais on y gagne en temps de développement.

Nous allons présenter les choix effectués et les schémas de principes sous-jacents ; notre but n'est pas de rentrer dans les détails de l'implémentation.

5.2 Base logicielle

5.2.1 OpenInventor

Pour développer notre application, nous avons besoin d'une bibliothèque de fonctions permettant de gérer les objets 3D, de gérer leur affichage et de gérer quelques interacteurs 3D simple. Notre choix s'est porté sur la librairie *OpenInventor* [Wer95].

A l'origine, cette librairie était disponible sur station de travail Silicon Graphics sous Unix, mais elle a récemment été portée sur diverses plateformes et divers systèmes d'exploitation dont les stations Sun sous Solaris et les PC sous Windows. Nous parlerons essentiellement de l'utilisation que nous en avons fait sur Silicon.

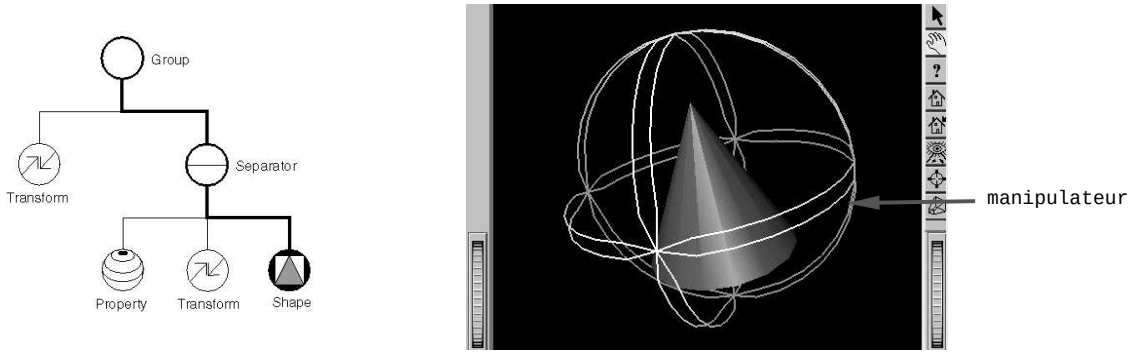

FIG. 5.1 – *OpenInventor*

OpenInventor met à disposition du programmeur un ensemble de fonctions accessibles en C++ (et en C) qui permettent de décrire et de gérer une scène 3D composée d'objets, de leur attributs (transformation géométrique, matière,...), de lumières et de caméra. Cette description se fait sous forme d'arbre.

OpenInventor permet d'afficher la scène à travers un visualiseur 3D. L'utilisateur peut se déplacer interactivement à l'intérieur de la scène; la scène est réaffichée en temps réel sous le nouveau point de vue. Ce visualiseur 3D est, en fait, un "widget", c'est-à-dire un cadre qui sera utilisé dans la construction d'une interface utilisateur comportant d'autres éléments (les widgets) comme des boutons, des menus déroulants, etc...

Enfin, OpenInventor met à la disposition du programmeur un ensemble de

manipulateurs 3D. Ils sont représentés dans le visualiseur sous formes de poignées sur lesquelles l'utilisateur pourra agir par l'intermédiaire de la souris. En fonction de cette action, une transformation particulière pourra être appliquée à un objet (translation, rotation,...). C'est la base de l'interaction entre l'utilisateur et la scène.

5.2.2 Motif et Tcl

Comme nous l'avons vu, le visualiseur d'OpenInventor n'est qu'une composante de l'interface graphique utilisateur. Son utilisation n'est pas suffisante pour construire une application complète, il faut aussi décrire les éléments que l'on trouve traditionnellement dans les applications interactives : les boutons, le menus, les champs d'entrée textuelle, etc...

Pour cela, nous avons utilisé *Motif*. Cette librairie permet de créer et de gérer ces divers éléments et est devenue un standard dans l'industrie. De plus, le visualiseur 3D d'OpenInventor est directement intégrable dans une interface Motif. Par contre, elle reste complexe à utiliser par le programmeur. L'appel aux fonctions Motif se fait à travers un programme écrit en C ou en C++. Même si cela ne pose pas de problème majeur, la mise au point d'une interface graphique nécessite beaucoup de tâtonnement pour placer correctement les divers éléments de l'interface et gérer leur interactions. Cette phase nécessite de nombreuses recompilations du programme avant d'obtenir satisfaction. Cela devient très vite fastidieux.

Pour y remédier, nous avons aussi utilisé un langage de commande interprété *Tcl* [Ous93] dont l'une des extensions[1] nommée *TclMotif* permet de construire et de piloter une interface graphique Motif [GN94]. Le principal intérêt est de ne pas avoir à recompiler le programme pour modifier l'interface puisque le langage est directement interprété. Bien sûr, on y perd en temps d'exécution, mais ceci ne concerne que le temps de réponse à une sollicitation de l'utilisateur via l'interface, qui reste tout de même quasi-instantanée.

TclMotif a un autre avantage : il est possible de lui rajouter de nouvelles fonctionnalité facilement. Il suffit pour cela d'écrire une fonction en C ou C++ et de l'associer à une nouvelle commande du langage Tcl. Ainsi, dans un script Tcl, on pourra appeler la nouvelle fonctionnalité.

1. une extension plus connue est Tcl/Tk dont l'interface graphique utilise directement Xlib

5.2.3 L'application

Nous avons utilisé le principe de l'ajout de nouvelles fonctionnalité au langage TclMotif pour y intégrer nos propres fonctions. Ce qui nous a permis de développer un modeleur interactif dédié à nos outils de déformation et à notre modèle de type fusion/flexion.

Le schéma de principe décrivant les différentes couches de l'application correspond à la figure 5.2.

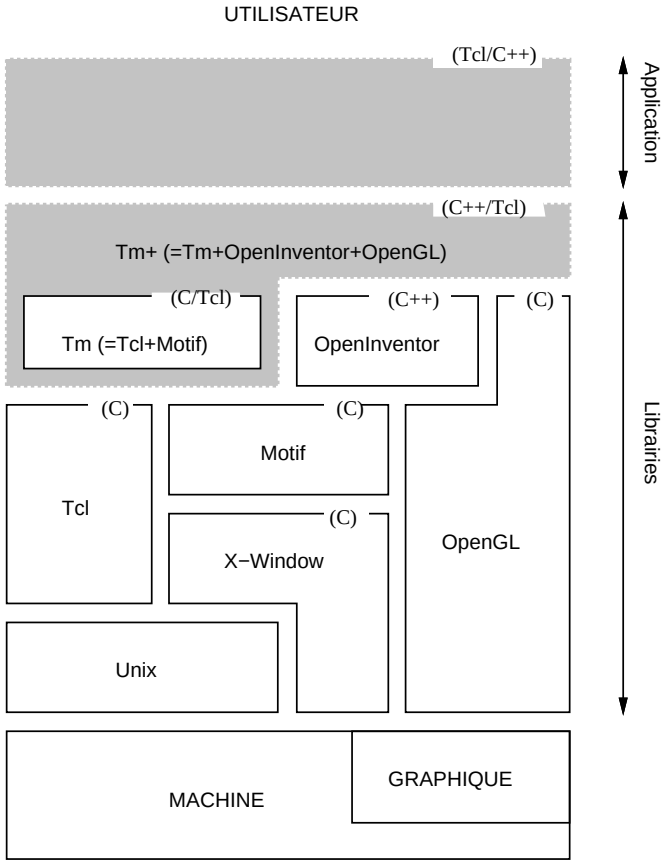

FIG. 5.2 – *Du cœur de la machine à l'utilisateur : les différentes couches logicielles traversées (en gris, les couches que nous avons développées)*

La couche *Tm+* rajoute à la couche TclMotif :

- les visualiseurs 3D d'OpenInventor accessibles sous forme de widget Motif ;
- quelques fonctionnalités de dessins issues d'OpenGL [NDW95] (librairie graphique de dessin 2D et 3D, OpenInventor utilise cette librairie pour effectuer

ses affichages temps réel) ;

- la gestion des objets 3D :
  - les fonctions de création,
  - les fonctions de déformation (fusion et flexion),
  - les fonctions de visualisation (qui font appel à OpenInventor) ;
- la gestion des outils de déformation :
  - les fonction de création,
  - les fonction de visualisation de l'outil (via OpenInventor),
  - les fonction de manipulation (qui font appel aux manipulateurs 3D d'OpenInventor),
  - liaison entre la manipulation de l'outil de déformation et la fonction de déformation appliquée à l'objet.
- la gestion des outils de transformation plus classiques (translation, rotation, homothétie,...)

Ces fonctions sont écrites en C++ et sont appelées ensuite à travers un script Tcl de la manière suivante :

```
Cube C          # créons un objet cube nommé C
FusionSph F     # créons un outil de type fusion de forme sphérique F
C DEF deform F  # C est déformé par l'outil de déformation F
```

Dans cet exemple, le cube C est déformé par un outil de forme sphérique F. Nous avons implémenté cette opération de telle sorte qu'elle se réactualise automatiquement. Ainsi, si, par la suite, la forme, la position ou le centre de l'outil F est changé, la déformation appliquée à C sera automatiquement recalculée. Le changement effectué sur F peut l'être via une commande Tcl, mais aussi par l'utilisateur via un manipulateur 3D. Ainsi, l'utilisateur voit directement l'effet de ses manipulations sur l'objet déformé, et cette fonctionnalité est facile à gérer pour le programmeur.

La couche *application* est écrite essentiellement en Tcl. Elle décrit l'interface graphique : comment sont placés les widgets les uns par rapport aux autres. Elle décrit aussi la gestion des événements : ce qui se passe si l'utilisateur active tel bouton ou s'il agit sur tel manipulateur.

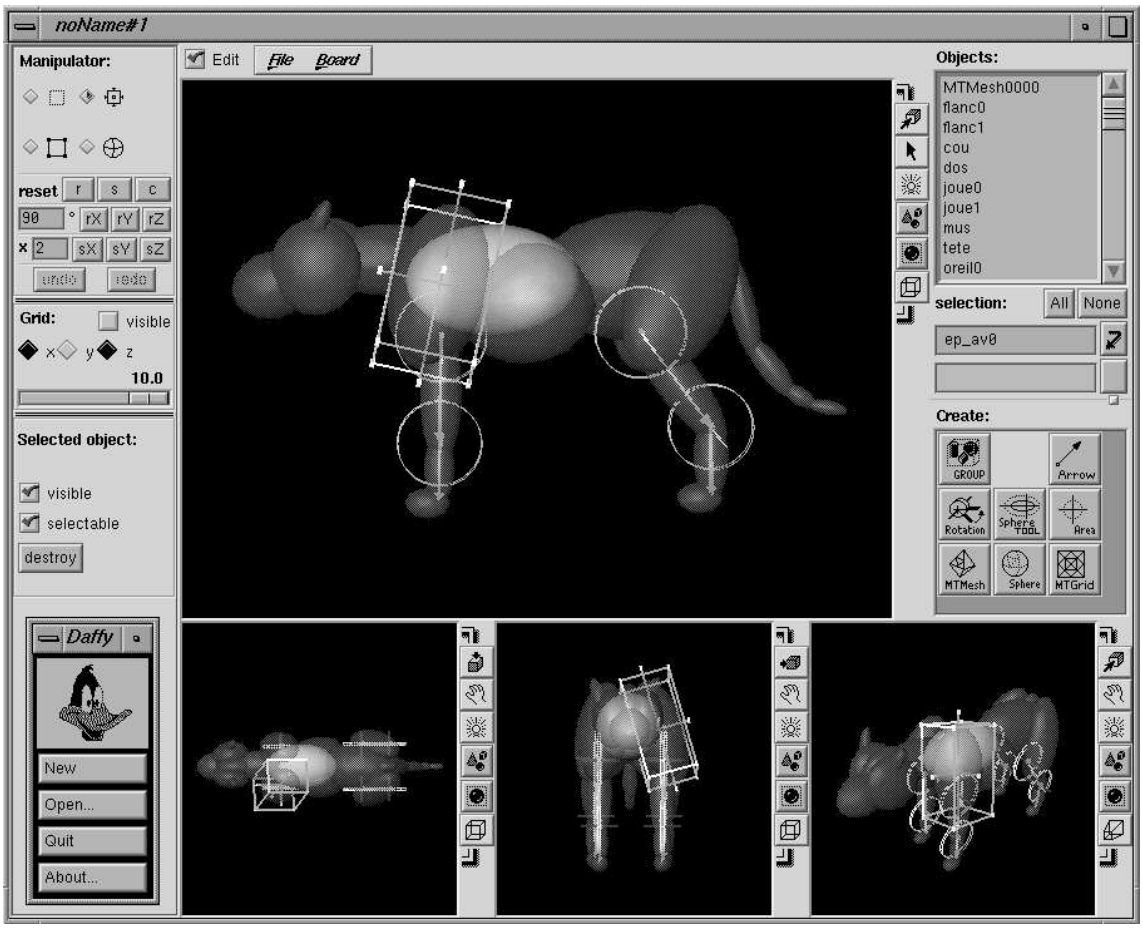

FIG. 5.3 – *Vue d'ensemble de l'interface graphique utilisateur*

C'est donc cette couche qui "pilote" toute l'interface (figure 5.3). Nous allons maintenant nous intéresser à ce que fait cette interface.

5.3 Fonctionnalités

Dans cette section, nous allons montrer les principales fonctionnalités de notre application à travers une série de captures d'écrans.

– Figure 5.4 : Après avoir créé un objet de base, l'utilisateur commence à le déformer. Il lui applique une déformation de type fusion avec un outil de forme sphérique. L'utilisateur n'a pas spécifié de zone d'influence, la déformation est donc appliquée à tout l'objet.

- Figure 5.5 : L'utilisateur déforme l'objet par un outil de type flexion dans le but de créer une articulation à cet endroit. Il spécifie la zone d'influence et peut agir sur la valeur de l'angle de flexion pour voir l'effet obtenu.

- Figure 5.6 : Une nouvelle déformation de type fusion est appliquée. La forme de l'outil est ellipsoïdale et le centre de l'outil est placé au niveau de l'articulation. Le but est de modeler la partie articulée de l'objet. De plus une zone d'influence est définie, assurant ainsi que l'objet ne sera pas déformé en dehors de cette zone.

- Figure 5.7 : L'utilisateur peut alors raffiner plus ou moins le maillage de l'objet qu'il est en train de créer. En fait, lors des deux premières fusions, le maillage a déjà été raffiné très légèrement sans l'intervention de l'utilisateur, ceci afin de voir grossièrement quelle sera la forme obtenue. Ici l'utilisateur spécifie une zone d'intérêt et décide d'appliquer l'algorithme de raffinement dans cette zone. Il contrôle pour cela les deux critères utilisés par l'algorithme et basés sur la longueur des arêtes et sur la courbure locale (cf. section 4.5). Il affine ainsi ces critères jusqu'à obtenir le niveau de détail désiré ; à chaque nouvelle valeur de critère, le maillage correspondant est généré.

- Figure 5.8 : Une fois le maillage raffiné, l'utilisateur a sous les yeux le résultat.

- Figure 5.9 : Il peut alors associer une texture à l'objet sous forme d'une image 2D qui sera plaquée sur l'objet. L'utilisateur choisi pour cela une position de l'objet (correspondant à une valeur d'angle de flexion), puis il plaque la texture sur l'objet en utilisant une méthode de projection (ici, il a utilisé une projection cylindrique). Lorsqu'il modifie la valeur de l'angle de flexion, la texture suit les déformations subies par l'objet.

Ces actions représentent les étapes clefs de la création d'un modèle articulé basé sur nos outils de fusion et de flexion. Ici, une seule articulation a été créé ; pour obtenir un modèle tel que le tigre de la figure 4.6 il faut répéter ces étapes plusieurs fois. On obtient ainsi un modèle "animable", c'est-à-dire un modèle auquel est associé un faible nombre de paramètres correspondant aux angles de ses articulations. En modifiant un angle, le modèle se modifie en conséquence. Donc, en faisant varier les paramètres habilement et continûment au cours du temps, on obtient un modèle animé (par exemple, un tigre en train de marcher).

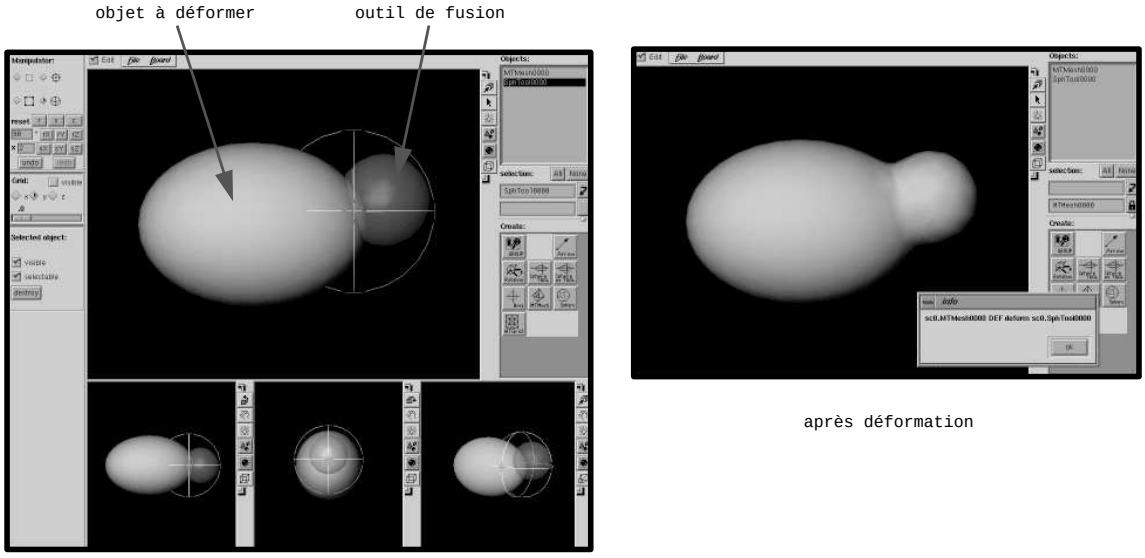

FIG. 5.4 – *Outil interactif : simple déformation par fusion avec un outil sphérique*

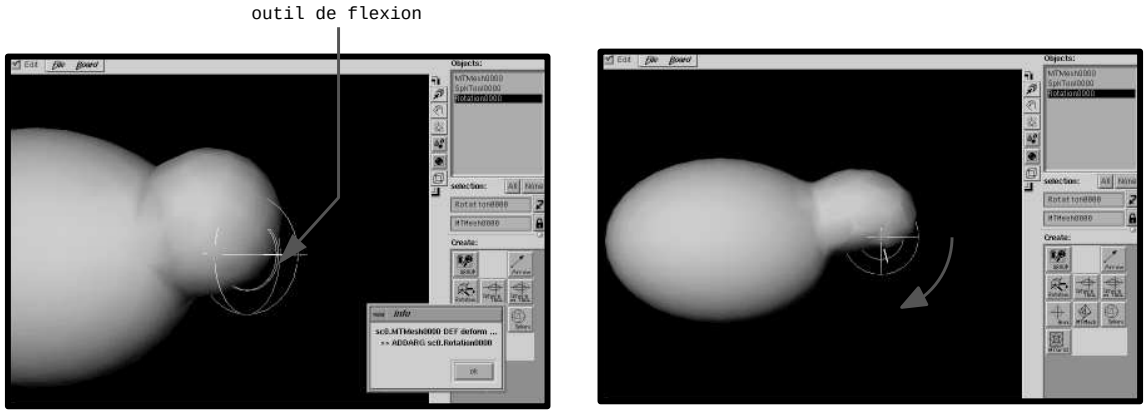

FIG. 5.5 – *Outil interactif : déformation de type flexion*

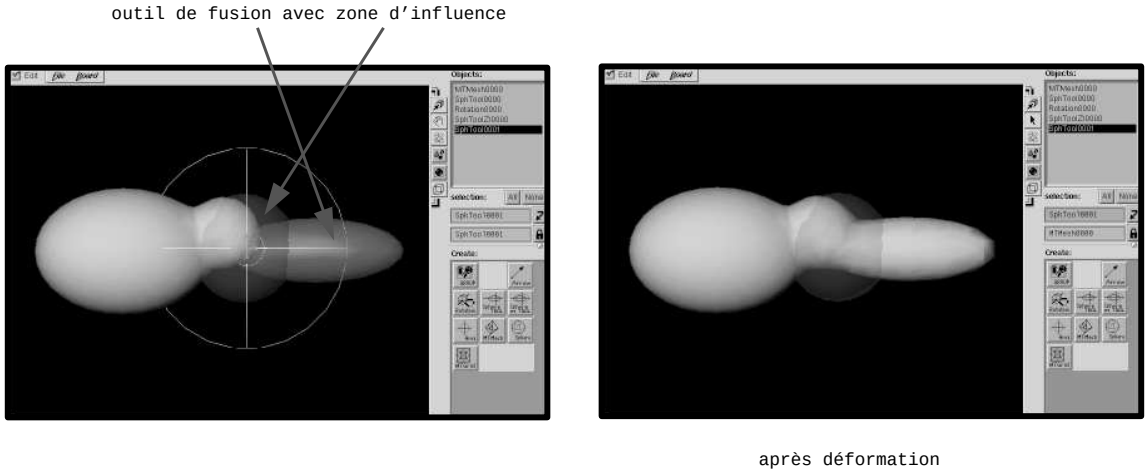

FIG. 5.6 – *Outil interactif : déformation par un outil de fusion ellipsoïdal muni d'une zone d'influence*

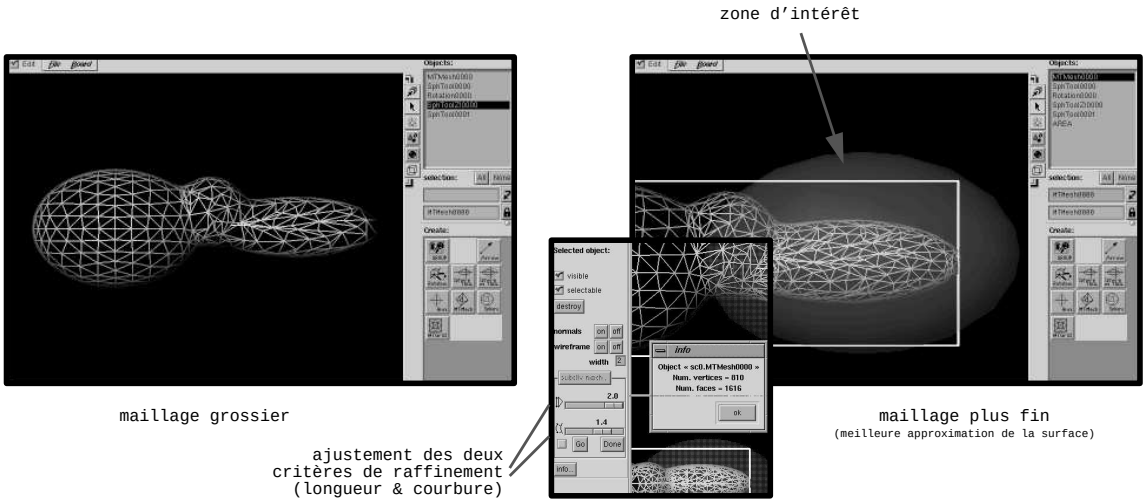

FIG. 5.7 – *Outil interactif : le maillage correspondant à la surface de l'objet déformé est raffiné à l'intérieur d'une zone d'intérêt ; l'utilisateur ajuste interactivement les critères de l'algorithme de raffinement*

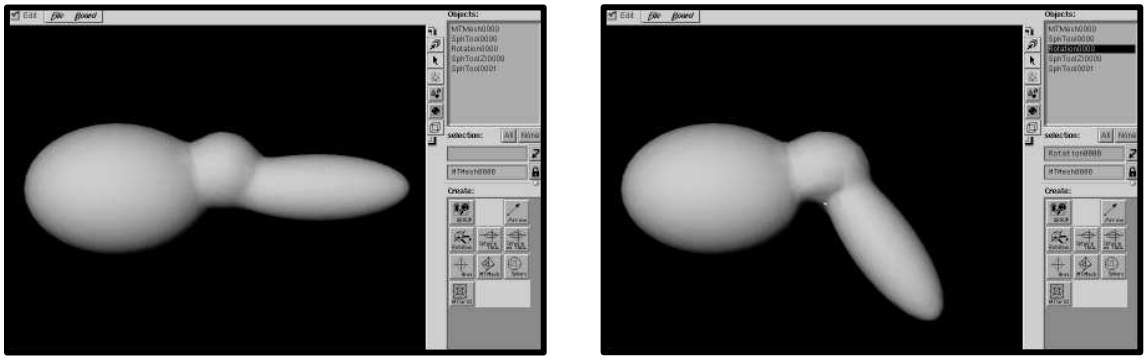

FIG. 5.8 – *Outil interactif : l'objet obtenu est interactivement animé en agissant sur la valeur de l'angle de flexion*

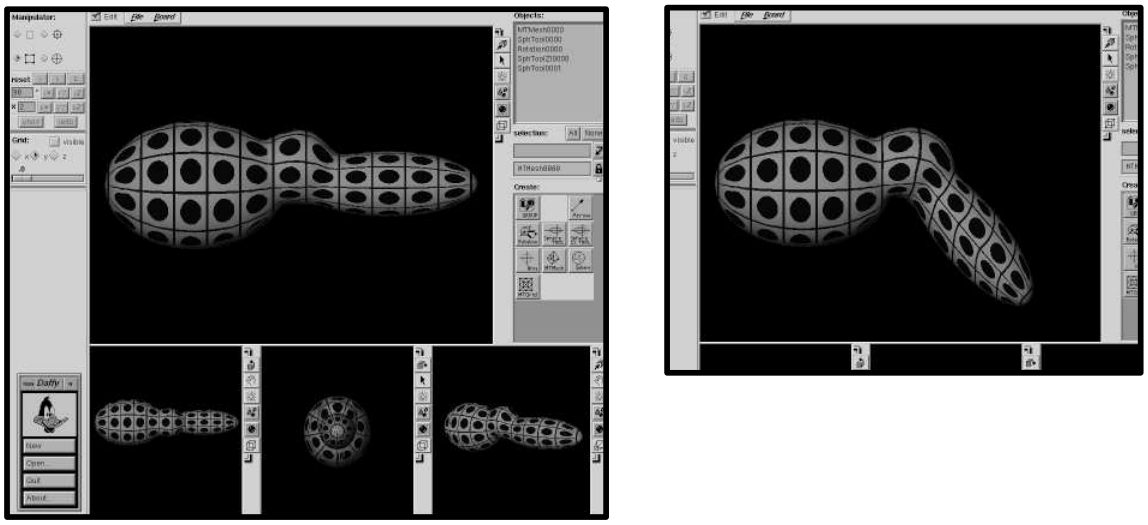

FIG. 5.9 – *Outil interactif : une texture est plaquée sur l'objet*

Chapitre 6

Conclusion sur la modélisation par fusion de formes

6.1 Apports aux techniques de modélisation

- Les outils de déformation

Nous avons présenté deux nouveaux outils de déformation. Ces déformations sont des déformations de l'espace et s'appliquent donc à tous types d'objets indépendemment de leur topologie et de leur représentation.

- L'outil de fusion permet de déformer un objet en le fusionnant avec une forme 3D simple. L'objet déformé englobe alors la forme simple ou est creusé par elle.
- L'outil de flexion permet de tordre un objet autour d'un axe. L'objet déformé se comporte alors comme un objet souple articulé.

La principale caractéristique de ces déformations est de conserver la continuité. Ainsi, un objet dont la surface est lisse gardera cet aspect après déformation. Ces déformations sont donc de bons outils pour modeler des formes non polyédriques (sans arêtes vives).

Ces outils peuvent agir plus ou moins localement. L'utilisateur peut définir une zone à l'intérieur de laquelle l'objet subira la déformation. L'utilisateur contrôle ainsi l'influence de la déformation tout en s'assurant d'une transition douce entre la partie de l'objet déformée et celle qui ne l'est pas.

Le contrôle de la déformation peut se faire de différentes façons.

Pour les objets déformés par l'outil de fusion, il est possible de garder le contrôle du volume de l'objet déformé. La déformation est contrôlée par la forme de l'outil, sa position, la position de son centre de fusion, et par la zone d'influence de la déformation.

Pour les objets déformés par l'outil de flexion, la déformation est contrôlée par la position et l'orientation de l'outil, par l'angle de la flexion, et par la zone d'influence de la déformation.

Ces déformations ne modifient pas la topologie de l'objet déformé.

Elles s'appliquent facilement aux objets représentés par un maillage de leur surface. Le maillage peut être raffiné pour mieux approximer la déformation.

L'utilisation de cette représentation en facettes permet de calculer la déformation très rapidement. Ainsi, lorsque nos outils sont utilisés dans un modeleur graphique, l'utilisateur peut les manipuler interactivement et se rendre compte de leur effet immédiatement. Ceci contribue à rendre intuitive leur utilisation.

- Le modèle dédié

Les outils de déformation par fusion et par flexion sont particulièrement bien adaptés à la modélisation de formes d'aspect lisses. Ceci nous a conduit à proposer un modèle permettant de représenter des formes de type organique (tels que les corps humains ou d'animaux).

Ce modèle est constitué d'une forme de base et d'une succession de déformations de type fusion ou flexion. Ces déformations seront appliquées successivement à la forme de base pour obtenir la forme de l'objet final. Un maillage de la surface de l'objet s'obtient directement à partir du modèle. Les fusions seront utilisées pour créer les membres et les muscles, et les flexions seront utilisées pour créer les articulations.

L'objet est construit interactivement. L'utilisateur déforme progressivement l'objet en utilisant des fusions ou des flexions. Le maillage de l'objet est alors déformé en conséquence et l'utilisateur voit immédiatement le résultat de la déformation. Il décide alors de raffiner localement le maillage si besoin est. Puis, il a la possibilité d'appliquer une texture sur l'objet.

L'objet peut être animé. L'animation se fait en agissant sur les paramètres des déformations (les angles des flexions par exemple). L'objet s'anime en se déformant de telle sorte que le maillage de l'objet et sa texture suivent correctement l'animation.

Le maillage de l'objet peut s'obtenir à différents niveaux de détails. Un maillage plus ou moins fin de la surface de l'objet est obtenu en agissant sur les paramètres de l'algorithme raffinement de maillage.

Le modèle permet donc de décrire des formes de type organique, et de gérer le maillage de sa surface. Cette gestion se fait aussi bien sur le plan du raffinement local et global du maillage (niveaux de détail) que sur le plan de son comportement lors de l'animation du modèle.

6.2 Perspectives

Le modèle proposé dans cette thèse peut être étendu en y intégrant de nouveaux outils de déformations. Il pourrait aussi gérer d'autres aspects comme les changements topologiques, l'animation ou encore les métamorphoses.

- Etendre les outils de déformation

On peut imaginer de nouveaux d'outils de déformation qui pourront être intégrés dans notre modèle. La déformation doit s'exprimer sous la forme d'une déformation continue de l'espace. Pour pouvoir être utilisée pour déformer localement un objet, il faut en plus qu'elle puisse prendre en compte une zone d'influence.

Une extension utile à notre modèle serait une déformation de l'espace permettant de créer des formes tubulaires sur un objet (cou de girafe, queue du chat, tentacules de pieuvre,...). Il serait ensuite possible d'obtenir des effets comparables à ceux obtenus par les déformations axiales décrites en 1.3.2 pour les déformer et les animer. Il suffirait de spécifier un axe à l'intérieur de l'objet et de déformer cet axe.

- Changement topologiques

On peut imaginer des déformations de l'espace qui créent un changement de topologie sur l'objet déformé. Pour cela, il faut définir une déformation qui puisse passer continûment d'une topologie à une autre ; par exemple, d'une sphère à un tore. Ce n'est pas une chose facile, d'autant plus qu'il y a plusieurs façons d'effectuer cette transformation [DG96].

- Animation

Nous avons défini un modèle qui permet de créer des objets dont la forme peut évoluer lorsqu'on agit sur certains paramètres, afin d'animer ces formes. On pourrait penser à intégrer dans le modèle lui-même un ensemble d'animations génériques, c'est-à-dire un ensemble de façons de faire varier les paramètres au cours du temps pour obtenir une animation type. Par exemple, si on a modélisé un chat, le modèle pourrait inclure une animation de type "marche" qui agirait sur les articulations des pattes du chat afin simuler le chat en train de marcher.

Ce type d'extension faciliterait l'utilisation de tels modèles pour la production de scènes animées.

- Métamorphoses

Partant de deux objets représentés par notre modèle, on peut envisager de créer une métamorphose de l'un vers l'autre.

Supposons que les deux objets sont définis par la même succession d'outils où seuls les paramètres définissant les outils changent (par exemple, un chat et un tigre). La métamorphose de l'un vers l'autre peut se faire simplement en interpolant ces paramètres (ce qui revient à "interpoler" les outils).

Si les deux objets ne sont pas définis par le même nombre et la même succession d'outils, il est plus difficile de concevoir un processus de métamorphose complètement automatique. On pourrait envisager un processus semi-automatique où l'utilisateur spécifie quel outil se transforme en quel autre. Il faudrait alors que le programme se charge de gérer les interpolations, l'ordre des déformations, et comment apparaissent ou disparaissent les outils supplémentaires.

Toutes ces extensions doivent permettre d'obtenir un modèle vraiment adapté aux manipulations qu'un graphiste est en droit d'attendre d'un logiciel de modélisation, et aussi adapté aux utilisations qu'il peut faire des objets ainsi créés dans le cadre des applications de type audiovisuelle ou réalité virtuelle.

Rendu non-photoréaliste

Chapitre 7

Rendu de scènes 3D imitant le style «dessin animé»

7.1 Introduction

Après avoir parlé de modélisation, nous allons nous intéresser ici à un autre aspect de la synthèse d'image : le rendu non-photoréaliste.

La technique présentée ici a pour but de produire des images de synthèse ayant un style «dessin animé» à partir de scènes tridimensionnelles.

Contrairement aux images de synthèse traditionnelles où l'objectif est d'obtenir un rendu aussi proche que possible de la réalité (photoréalisme), les images de dessins animés utilisent un rendu plus épuré et plus suggestif [Duc83]. Ainsi, les personnages et les objets présents sur l'image sont caractérisés par leur contour, l'intérieur du contour étant généralement rempli par une couleur unie (voir figure 7.1).

Les travaux visant à produire des images non-photoréalistes sont assez recents.

G. Winkenbach et D. Salesin [WS94] ainsi que M. Salisbury et al. [SABS94] se sont intéressés aux illustrations à l'encre et à la plume. L'effet obtenu et assez proche du style bande dessiné, mais la méthode utilisée est limité. En effet, dans [WS94], seules les objets composées de surfaces planes peuvent être traités. G. Winkenbach et

D. Salesin viennent de proposer une extension [WS96] permettant de traiter aussi les surfaces paramétriques. Dans [SABS94] le processus est interactif, donc difficilement utilisable pour générer une animation.

B. Meier [Mei96] propose un algorithme qui permet de générer des animations en imitant le style que peut adopter un peintre pour dessiner un tableau. Toute la difficulté est de faire en sorte que ce style de rendu, imitant les coups de crayons et de pinceaux, reste cohérent pendant l'animation. Ainsi, il ne faut pas que l'on ai l'impression que la direction des tracés varie aléatoirement entre chaque image.

Notre algorithme utilise un ensemble de techniques plus ou moins classiques en image de synthèse. Ainsi, le calcul des ombres utilise la technique des «shadow maps» [Wil78] et le tracé des contours s'inspire de techniques de traitement d'images décrites par Saito et Takahashi [ST90]. L'originalité de notre travail vient principalement de la façon dont ces techniques sont adaptées et mixées ensembles.

7.1.1 Aspects caractéristiques

Il existe différents niveaux de qualité de dessin. Les contours peuvent être plus ou moins appuyés, les aplats de couleur peuvent inclure des dégradés, les effets de lumière et d'ombre peuvent créer une ambiance... Nous ne cherchons pas ici à reproduire absoluement tous les effets possibles. Nous souhaitons simplement obtenir des images ayant clairement un style «dessin animé» ou «bande dessinée». Pour cela, nous nous attacherons à produire des contours d'épaisseur constante et des aplats de couleur unis. Nous ajoutons à cela trois effets supplémentaires : la possibilité de remplacer la couleur unie des aplats par une texture, l'ajout de taches spéculaires (reflet de la source) sur certains objets, et les ombres.

- La texture n'est pas indispensable, mais elle permet d'enrichir facilement un dessin (pour obtenir, par exemple, un motif répétitif sur une surface).

- Les taches spéculaires permettent de caractériser la matière d'un objet (brillant, métallique,...). Elles correspondent aux reflets des sources de lumière sur l'objet et sont représentées par des taches très localisées de couleur claire (voisine du blanc).

- Quant aux ombres, elles ajoutent considérablement à la compréhension de la scène et à la perception de la 3D. On peut distinguer deux types d'ombres: les ombres propres (ou auto-ombres) correspondant aux côtés non éclairés des objets, et les ombres portées correspondant aux zones occultées par un objet

intercalé entre la zone et la source de lumière. Ces deux aspects feront l'objet de traitements distincts. Dans les dessins animés classiques «bons marchés», les ombres ne sont généralement pas représentées. Dans les dessins animés plus élaborés, les ombres propres sont figurées. Une zone d'un objet à moitié dans la lumière et à moitié dans l'ombre aura alors deux couleurs : la couleur de l'objet pour la partie éclairée, et cette même couleur assombrie pour la partie non éclairée. La frontière entre les deux est franche, contrairement aux images de synthèse traditionnelles où la transition est progressive (en dégradé). Dans le cas d'une source de lumière unique et immobile, ces ombres propres sont relativement simples à dessiner à la main, même si l'objet est en mouvement, car elles restent relativement localisées au cours d'une animation, surtout si la lumière est directionnelle (source à l'infini). Par contre, les ombres portées d'objets en mouvement peuvent évoluer beaucoup lors d'une animation. Cela explique qu'on ne représente généralement que les ombres portées d'objets fixes, comme le décor.

FIG. 7.1 – *Théière rendue par un algorithme de synthèse classique (à gauche) et par notre algorithme «dessin animé» (à droite).*

7.1.2 Principe

Notre algorithme produit une image à partir d'une description géométrique de la scène 3D. Tous les types de représentation usuels peuvent convenir; de ce point de vue, l'algorithme est relativement général. Dans notre implémentation, nous utilisons, en entrée de l'algorithme, des scènes représentées sous forme d'arbre

ambiant, $diffus$ et *spéculaire* sont des vecteurs couleurs (de composantes Rouge, Vert, Bleu). La multiplication de deux vecteurs couleurs se fait en multipliant les composantes deux à deux. $\boldsymbol{n}$ est le vecteur normal à la surface au point considéré, $\boldsymbol{l_i}$ est le vecteur dirigé vers la source i, et $\boldsymbol{s_i}$ est le vecteur médian entre $\boldsymbol{l_i}$ et le vecteur dirigé vers la caméra.

Dans la suite, nous allons appliquer ce modèle de façon à obtenir les différents effets désirés.

7.2 L'algorithme

7.2.1 Vue d'ensemble

La figure 7.3 présente les différentes étapes de notre algorithme de rendu. Chaque étape produit une image, la combinaison de ces images donne l'image finale.

Ces étapes que nous détaillerons par la suite sont :

- rendu de la scène en lumière ambiante ;
- calcul des contours des objets de la scène, obtenu à partir du Z-buffer et du buffer des normales ;
- pour chaque source de lumière :
  - rendu de la scène éclairée par cette seule source de lumière ;
  - calcul des ombres propres et des ombres portées dues à cette source, obtenues à partir du Z-buffer associé à la source, du rendu éclairé par cette seule source et du buffer des normales ;
- combinaison des images.

7.2.2 Les différentes images et buffers calculés

Les images et buffers nécessaires pour chaque étape de calcul sont les suivant :

- Rendu de la scène en lumière ambiante. Cette image est produite en calculant un rendu de la scène vue de la caméra en ayant préalablement éteint les sources de lumière. Chaque objet a alors une couleur ou une texture sans effet d'illumination (les couleurs sont unies), atténuée par l'intensité ambiante de la scène. Par exemple, si l'ambiant vaut (0.2, 0.2, 0.2), un objet rouge (R=1, V=0, B=0) apparaîtra uniformément rouge foncé (R=0.2, V=0, B=0).
- Z-buffer de la scène vue de la caméra. On profite du calcul de l'image précédente (rendu ambiant) pour récupérer son Z-buffer (i.e. carte de profondeur, en chaque point est stocké la distance du point 3D à la caméra).
- Buffer des normales associées à la scène. En chaque point de ce buffer sont stockées trois valeurs réelles : les coordonnées de la normale associée au point 3D correspondant. Ce buffer de normales peut être obtenu en se ramenant à des calculs de rendu ordinaires pouvant être effectués facilement en utilisant OpenGL. Pour cela, nous calculons deux rendus de la scène vue de la caméra (figure 7.4). Pour ces deux rendus, les matières et textures des objets sont remplacées par une matière parfaitement diffuse et uniformément blanche (ambiant=noir, diffus=blanc (R=1, V=1, B=1), pas de spéculaire). Les sources de lumière sont éteintes. Le premier rendu (appelons le I_1) est obtenu en

ajoutant trois sources de lumière directionnelles (sources à l'infini) à la scène respectivement de couleur rouge et de direction $+\mathbf{x}$, de couleur verte et de direction $+\mathbf{y}$, et de couleur bleue et de direction $+\mathbf{z}$. Le calcul d'illumination en un point où la normale vaut $\boldsymbol{n} = (n_x, n_y, n_z)$ donne :

$$couleur = rouge \times \max\{\boldsymbol{n}.\boldsymbol{x}, 0\} + vert \times \max\{\boldsymbol{n}.\boldsymbol{y}, 0\} + bleu \times \max\{\boldsymbol{n}.\boldsymbol{z}, 0\} \ .$$

La couleur du point vaut donc $\max\{n_x, 0\}$ dans la composante rouge, $\max\{n_y, 0\}$ dans la composante verte et $\max\{n_z, 0\}$ dans la composante bleue. Le deuxième rendu (I_2) est obtenu en inversant la direction des trois sources directionnelles : la rouge suivant **-x**, la verte suivant **-y**, la bleue suivant **-z**. La couleur du point vaut alors :

$$couleur = rouge \times \max\{-n_x, 0\} + vert \times \max\{-n_y, 0\} + bleu \times \max\{-n_z, 0\} \ .$$

Finalement, il suffit de calculer $I_1 - I_2$ pour obtenir un buffer dans lequel les composantes rouge, vert, bleu correspondent respectivement à n_x, n_y, n_z.

- Pour chaque source de lumière :
  - Rendu de la scène éclairée uniquement par cette lumière. Pour ce rendu, on ne tient pas compte de l'ambiant global de la scène. On souhaite calculer une image sans véritable calcul d'illumination afin d'obtenir les aplats de couleur sans dégradé qui caractérisent le coloriage des dessins animés ou BD classiques. Pour cela, il faut éliminer le terme en $\boldsymbol{n}.\boldsymbol{l}$ ($\boldsymbol{n}$ est la normale à l'objet, $\boldsymbol{l}$ est le vecteur dirigé vers la source) dans l'équation d'illumination de Phong. Si l'objet est spéculaire, on conservera tout de même la tache spéculaire car elle renseigne sur la nature de la matière de l'objet (mat, brillant, métallique,...). On effectue donc un rendu classique en ayant au préalable modifié la matière des objets de la façon suivante: on affecte à la couleur ambiante de l'objet la valeur de sa couleur diffuse, on met la couleur diffuse à noir, la couleur spéculaire ne change pas. Ce rendu permet d'obtenir une image où chaque objet est uniformément coloré par sa couleur diffuse (modulo la texture et la tache spéculaire).
  - Z-buffer de la scène vue de la source de lumière. Ce buffer est obtenu en remplaçant la source de lumière par une caméra. Un rendu de la scène vue de cette caméra est calculé et son Z-buffer est, alors, récupéré. Ce buffer sera utilisé pour le calcul des ombres (détaillé plus loin) dues à cette source de lumière; la technique utilisée est celle dite des *shadow maps*.

7.2.3 Les étapes de l'algorithme

7.2.3.1 Les contours des objets

Deux types de contours nous intéressent : les profils (ou contours apparents) et les arêtes vives des objets.

Les profils correspondent aux lieux où le regard (demi-droite qui part du centre de la caméra vers la scène en passant par un pixel donné) est tangent à la surface des objets. Ce lieu peut être obtenu par des méthodes numériques, mais celles-ci sont généralement complexes à programmer car elles doivent gérer de nombreux cas particuliers et se révèlent souvent instables aux points où les profils s'interrompent à cause d'un changement de courbure de la surface (voir, par exemple, le patch bleu et blanc de la figure 7.3). Nous avons préféré utiliser une méthode plus simple et stable, mais un peu moins souple (inspirée de [ST90]). Les profils correspondent aussi aux discontinuités (d'ordre 0) présents dans le Z-buffer vu de la caméra. Il suffit donc d'appliquer un filtre *détecteur de contours* sur ce Z-buffer. Pour ce faire, nous utilisons l'opérateur différentiel d'ordre 1 de taille 3×3 suivant :

$$g = \frac{1}{8}\left(|A-x| + 2|B-x| + |C-x| + 2|D-x| + 2|E-x| + |F-x| + 2|G-x| + |H-x|\right)$$

où $A, B, \cdots, H$ sont les pixels voisins de x :

A	B	C
D	x	E
F	G	H

Cet opérateur permet d'obtenir le gradient de l'image en Z, il faut maintenant le seuiller pour obtenir les contours qui correspondent aux lieux de fort gradient. Le filtre non-linéaire 3×3 suivant est utilisé :

$$p = \min\left\{\left(\frac{g_{max} - g_{min}}{k_p}\right)^2, 1\right\}$$

où g_{max} et g_{min} sont les valeurs maximum et minimum du gradient dans le voisinage 3×3 considéré, et k_p est le seuil de détection compris entre 0 et 1. Plus k_p est petit, plus il y aura de contours détectés. Le choix de cette valeur est délicat; il s'agit d'un paramètre d'entrée de l'algorithme, il dépend principalement de la taille (largeur × hauteur) du Z-buffer car plus les variations de z sont précises, plus il est facile de dissocier une discontinuité d'une décroissance rapide mais continue de z. Dans notre implémentation, la valeur $k_p = 0.0001$ donne des résultats satisfaisant pour une image calculée à la résolution vidéo (768 × 576).

Les arêtes vives visibles des objets peuvent être détectées de façon similaire. En principe, un opérateur différentiel d'ordre 2 appliqué sur le Z-buffer devrait convenir puisque les arêtes sont des discontinuités de z d'ordre 1. Mais, pratiquement, cet opérateur se révèle complètement instable car beaucoup trop approximatif. Un filtre de plus grande taille devrait donner de meilleurs résultats, mais les calculs sont alors plus coûteux. Nous avons opté pour une autre solution: ces arêtes correspondent également à des discontinuités d'ordre 0 du buffer des normales. Il suffit donc d'appliquer à ce buffer un détecteur de contours, identique à celui proposé pour les profils, à la différence près qu'il faut considérer les trois composantes n_x, n_y, n_z de ce buffer dans le calcul du gradient. Le choix du seuil k_a (équivalent à k_p mais pour les arêtes) est moins sensible que celui de k_p, une valeur $k_a = 0.2$ donne de bons résultats.

Ces différents filtres nous permettent d'obtenir les contours caractérisant les objets de la scène de façon stable et en un temps très raisonnable. Il reste tout de même deux problèmes à résoudre : le contrôle de l'épaisseur des traits obtenus et le traitement de l'aliasing. L'épaisseur des contours est imposée par les deux passes de filtres 3×3, qui donnent au final un trait d'approximativement 5 pixels de large. Ce trait est aliasé, c'est la conséquence du seuillage assez abrupte que nous avons effectué : il évite d'avoir des traits flous, mais il génère de l'aliasing. Pour remédier à cet inconvénient, nous allons effectuer l'extraction des contours sur une image de résolution plus grande que la résolution finale. Le résultat sera rééchantillonné pour obtenir une image à la bonne taille. Ainsi, en effectuant le filtrage sur une image de taille double, les contours auront au final une épaisseur d'environ 2,5 pixels et seront de plus anti-aliasés puisqu'ils sont obtenus par moyennage des contours de 5 pixels de large. Bien entendu, plus on veut des contours fins et précis (anti-aliasés), plus les calculs sont lourds. Le calcul en taille double donne des résultats acceptables pour des images calculées à la résolution vidéo; les exemples présents dans ce chapitre (figures 7.1 et 7.6) sont calculés ainsi.

7.2.3.2 Les ombres

Pour chaque source de lumière, un rendu de la scène éclairée par cette seule source est calculé. Nous allons faire apparaître les ombres dues à cette source sur cette image.

Les ombres portées sont obtenues par la technique dite des «shadow maps». Cette technique consiste à vérifier pour chaque pixel si le point 3D correspondant est visible ou non de la source de lumière en utilisant le Z-buffer associé à la source

FIG. 7.2 – *Notations utilisées pour le calcul des ombres portées.*

En pratique, cet algorithme à un défaut majeur : il génère des ombres aliasées, car le Z-buffer associé à la source possède une résolution arbitraire qui peut s'avérer insuffisante par endroit. En effet, un objet occupant une grande place sur l'image vue de la caméra peut être petit sur l'image vue de la source. L'ombre portée par l'objet est alors imprécise (aliasée). Une astuce proposée par Reeves, Salesin et Cook [RSC87] permet de contourner ce défaut, voire même de le transformer en qualité puisqu'il permet d'obtenir des ombres douces (soft shadows). Il s'agit d'effectuer la comparaison en plusieurs pixels proches de $(x_l,\ y_l)$ et de moyenner le résultats (se reporter à l'article pour plus de détails).

Cet algorithme devrait permettre de détecter aussi les ombres propres des objets, mais le résultat n'est pas satisfaisant car, pour notre rendu «style dessin animé», la limite des ombres propres doit être franche. De plus, le z_s lu dans le Z-buffer de la source n'est généralement pas assez précis au niveau de cette limite d'ombre. Heureusement, il existe une autre façon d'obtenir les ombres propres des objets. L'ombre propre correspond à la partie d'un objet opposé à la lumière, c'est-à-dire la partie où $\boldsymbol{n}.\boldsymbol{l}$ est négatif ($\boldsymbol{n}$ est la normale au point considéré, $\boldsymbol{l}$ est le vecteur dirigé de ce point vers la source de lumière). Donc, le buffer des normales peut-être utilisé conjointement au Z-buffer de la caméra[1] pour effectuer ce test.

On obtient ainsi une image où apparaissent les ombres propres et portées dues à la source de lumière considérée. Le cumul des images pour chaque source permettra de déduire l'image finale, ainsi que nous allons le décrire maintenant.

7.2.3.3 Calcul de l'image finale

L'image finale produite par notre algorithme est obtenue en additionnant le rendu ambiant et toutes les images ombrées éclairées indépendamment par chacune des sources, puis en multipliant le résultat par {1 - buffer de contours}, c'est-à-dire un buffer valant 1 partout sauf sur les contours où il avoisine[2] 0. L'ambiant global de la scène, les sources de lumière et leurs ombres sont donc pris en compte, et les contours apparaissent en noir dans l'image.

7.3 Implémentation et résultats

Nous avons implémenté cet algorithme en C++ sur station Silicon Graphics. L'algorithme prend en entrée une scène décrite au format OpenInventor et utilise les librairies graphiques OpenInventor et OpenGL pour effectuer les différentes étapes de l'algorithme. Le principe consiste à utiliser ces librairies pour modifier la scène (matières et caméra) et à lancer des rendus *en mémoire uniquement* (offscreen rendering) pour obtenir les différents buffers.

Nous avons utilisé cet algorithme pour créer un petit dessin animé (1mn12) portant le doux nom de «Rendez-vous». Il est généré directement à partir de descriptions 3D des scènes qui le composent et de leurs animations. Il n'y a aucun dessin à la main. Ces scènes ont été modélisées en utilisant quelques outils standards

1. le Z-buffer permet de retrouver z_e, (x_w, y_w, z_w) est déduit de (x_e, y_e, z_e) et $\boldsymbol{l}$ est obtenu en normalisant le vecteur (position de la source - (x_w, y_w, z_w))

2. les contours sont anti-aliasés

sur station Silicon Graphics tels que *SceneViewer* (figure 7.5) et *Noodle*, et récupérées sous forme de scripts OpenInventor. L'animation est obtenue en éditant ces scripts et en ajoutant quelques noeuds animés dans l'arbre Inventor de description de la scène. La figure 7.6 montre quelques images extraites de ce film. Le film est composé en tout plus de 1500 images différentes calculées en résolution vidéo (768×576). Le calcul d'une image a pris en moyenne 6 mn sur une station INDY (MIPS R4400 200MHz).

7.4 Conclusion sur le rendu style «dessin animé»

Nous avons décrit un algorithme de rendu qui génère des images imitant le style «dessin animé» (ou «bande dessinée») à partir de la description tridimensionnelle d'une scène fixe ou animée.

L'effet «dessin animé» est obtenu en calculant différentes images et buffers à partir de la description de la scène et en les combinant. Ainsi, sur le rendu finale, les contours (profils et arêtes) des objets sont dessinés en traits noirs, les surfaces intérieures à ces contours sont colorés uniformément, et les ombres propres et les ombres portées apparaissent.

L'intérêt de notre méthode, par rapport aux techniques traditionnelles du dessin animé, est d'automatiser la production des images. Ceci permet de conserver la cohérence spatiale des objets dans la scène (mouvements de rotation d'objets ou de caméra, déplacements de sources lumineuses,...). De plus, les ombres aussi restent cohérentes.

Il reste, cependant, quelques extensions qu'il serait intéressant de développer. On aimerait, par exemple, pouvoir traiter des objets transparents, ou encore contrôler l'épaisseur des traits de contour indépendamment de l'anti-aliasing, et faire varier cette épaisseur en fonction de la courbure de la surface à cet endroit (technique fréquemment utilisée en bande dessinée pour appuyer la courbure).

FIG. 7.3 – *Schéma de principe*

FIG. 7.4 – *Construction du buffer des normales. L'image résultat (à gauche) comporte des valeurs R,V,B comprises dans* $[-1,+1]^3$ *correspondant à* n_x, n_y, n_z *(codage utilisé: -1,-1,-1=noir; 0,0,0=gris; 1,1,1=blanc).*

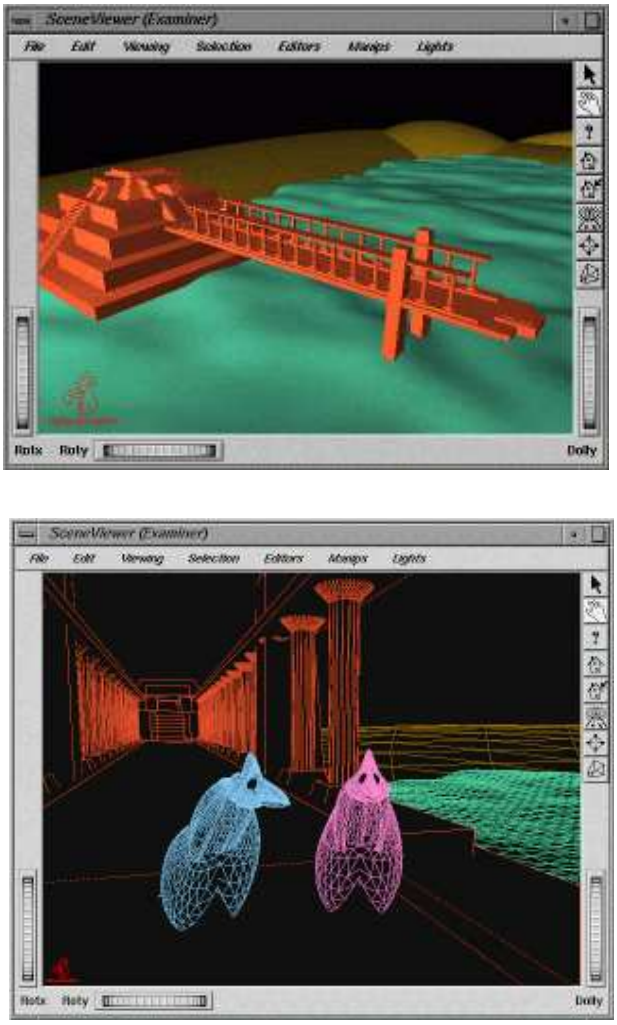

FIG. 7.5 – *Scènes Inventor utilisées pour «Rendez-vous»*

FIG. 7.6 – *Quelques images extraites du dessin animé «Rendez-vous»*

Annexes : réalisations

Annexe A

VRML : Virtual UnderWorld

Les personnes ayant accès à internet peuvent visualiser à distance une scène 3D. Pour cela, il suffit que la scène soit décrite en VRML (Virtual Reality Modeling Language) et de la rendre accessible sur le World Wide Web.

L' "internaute", qui a un visualiseur VRML, peut alors se déplacer à l'intérieur de la scène. Il peut aussi interagir avec elle. La plus simple des interactions consiste "clicker" à sur un objet de la scène pour accéder à une nouvelle scène.

Nous avons réalisé un petit monde VRML qui nous a permis de nous familiariser avec ce langage et les possibilités qu'il offre. Ce monde s'appèle "Virtual UnderWorld" et est accessible à l'adresse :

`http://www-rocq.inria.fr/syntim/underworld/`

L'originalité de ce monde est de permettre à d'autres d'y connecter leur propre monde VRML. Il est composé de différentes salles. La salle principale est un couloir. Les portes de ce couloir donnent accès à d'autres salles dont la majorité sont situées sur d'autres sites web.

Quelques captures d'écran de Virtual UnderWorld sont présentés sur les pages suivantes.

Avec l'arrivée de VRML 2.0, les interactions avec la scène peuvent être plus complexes et la scène peut être animée (il est même possible de définir les animations en java). Le mot "réalité virtuelle" commence à prendre vraiment son sens.

Dans ce cadre, il nous semble que l'utilisation du modèle par fusion/flexion présenté dans cette thèse pourrait être envisagé. Les objets représentés par ce modèle peuvent se déformer et s'animer. De plus, la gestion des niveaux de détails est particulièrement bien adaptée à ce genre d'utilisation : un objet éloigné sera maillé plus grossièrement qu'un objet proche.

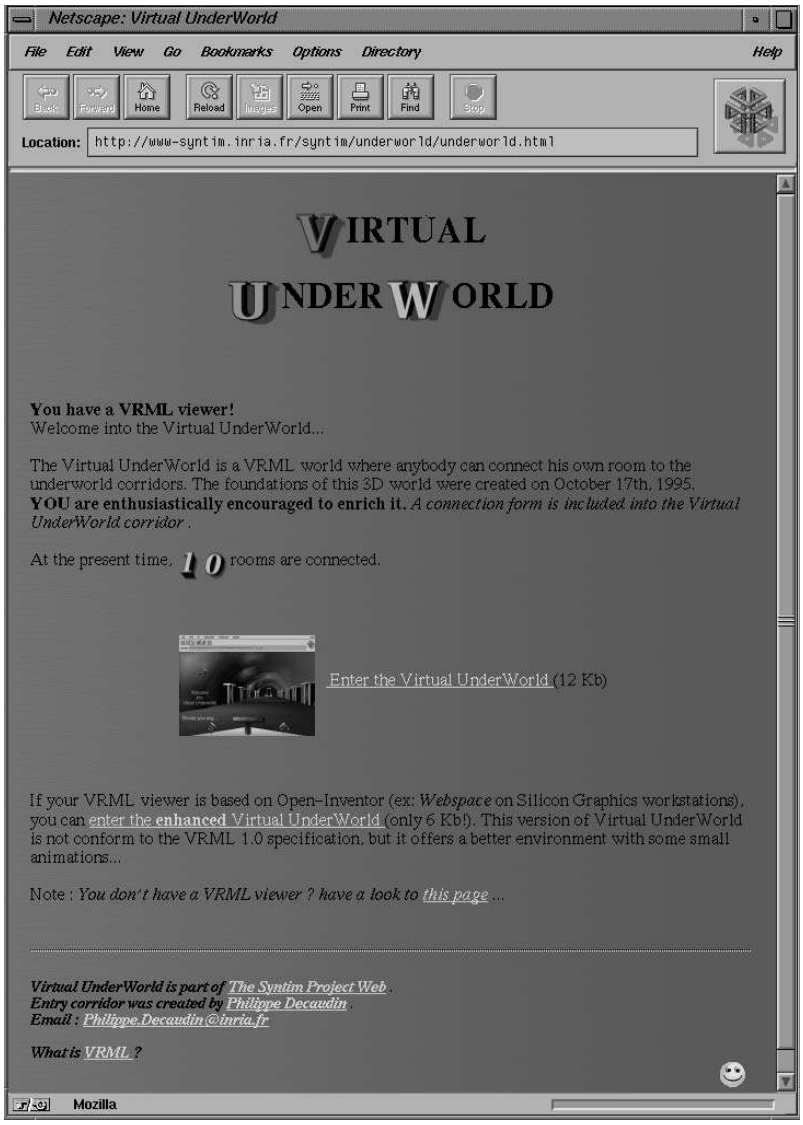

FIG. A.1 – *La page d'accueil de Virtual UnderWorld*

FIG. A.2 – *Virtual UnderWorld : le couloir donnant accès à différents sites VRML*

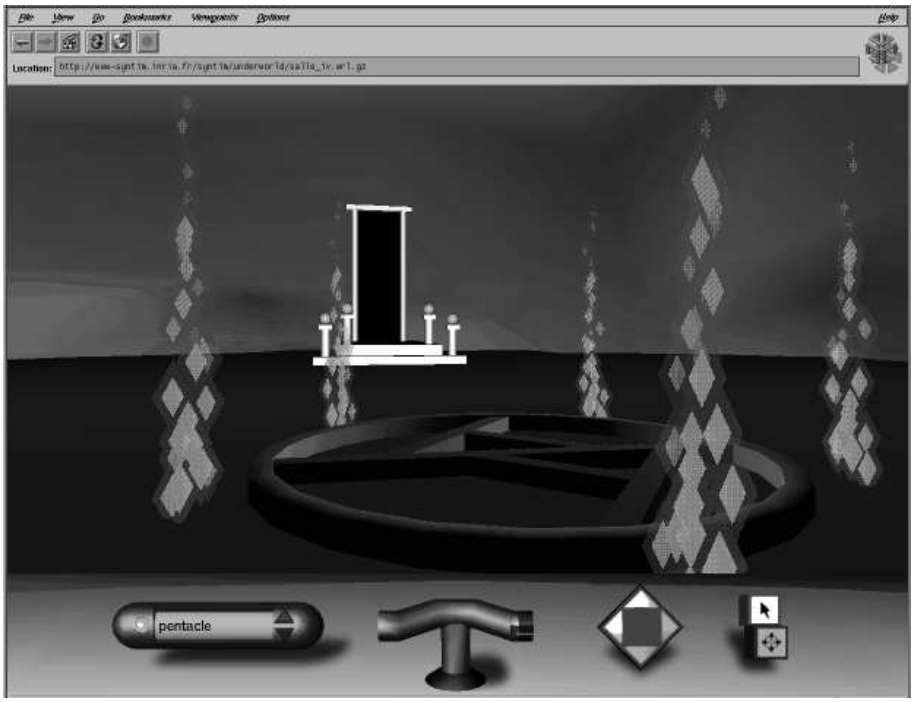

FIG. A.3 – *Une salle connectée au couloir d'UnderWorld*

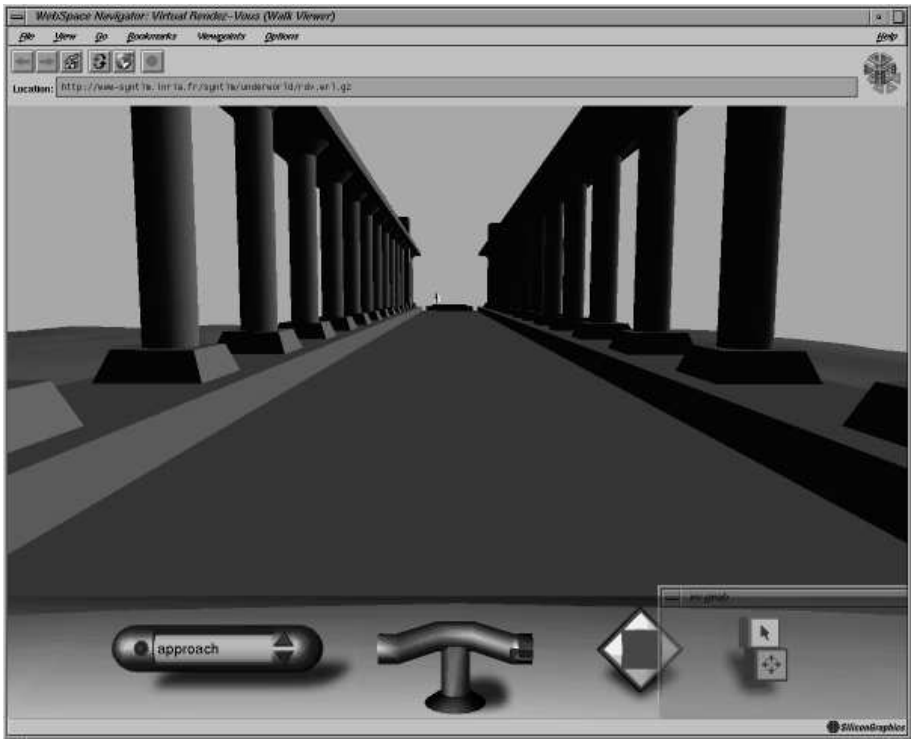

FIG. A.4 – *La scène 3D ayant servie pour le dessin animé "Rendez-vous" traduite au format VRML. Elle est accessible à partir d'UnderWorld*

Annexe B

Films en images de synthèse

B.1 Caverna Magica

Caverna Magica est un petit film en images de synthèse. Ce fut notre première réalisation.

Nous avons fait ce film à la fois dans un but artistique et dans un but démonstratif. Il nous a permis de montrer l'utilisation de la fusion d'objets décrite dans le chapitre 2. L'action se passe dans une grotte. On y voit des stalactites et des stalagmites grandir jusqu'à former une colonne. On y voit aussi quelques gouttes d'eau perler et tomber dans une flaque. Tous ces effets ont été obtenus en utilisant notre algorithme de fusion d'objets.

Nous avons utilisé le logiciel domaine publique de rendu par lancer de rayon POVRAY [Tea93]. La grotte a été modélisée par un script POVRAY. Les stalactites, stalagmites, colonnes et gouttes d'eau ont été modélisées en utilisant notre logiciel, les maillages obtenus ont ensuite été traduits en script POVRAY et inclus dans la scène.

Le rendu de chaque image au format vidéo (768×576) a nécessité environ 30 minutes sur Silicon Graphics Indigo2.

- **Contributions : Caverna Magica** ($\sim$ 2mn)
  Réalisation : *Philippe Decaudin, Arghyro Paouri*
  Montage/Son : *Christian Blonz*
  Production : *INRIA-France, 1994*

- *Caverna Magica* a été sélectionné et présenté aux manifestations suivantes :

  – F.A.U.S.T. 94 (Forum des Arts de l'Univers Scientifique et Technologique), Toulouse, France, 14-18 octobre 1994 ;

  – Effects and Animation, Computer Graphics Expo. Londre, Angleterre, 7 novembre 1994 ;

  – Computer Animation'95, Genèvre, Suisse, 19-22 avril 1995 ;

  – Concours A.C.I.E.R.S. 95 (Art et Culture dans l'Industrie et la Recherche Scientifique), a obtenu le troisième prix.

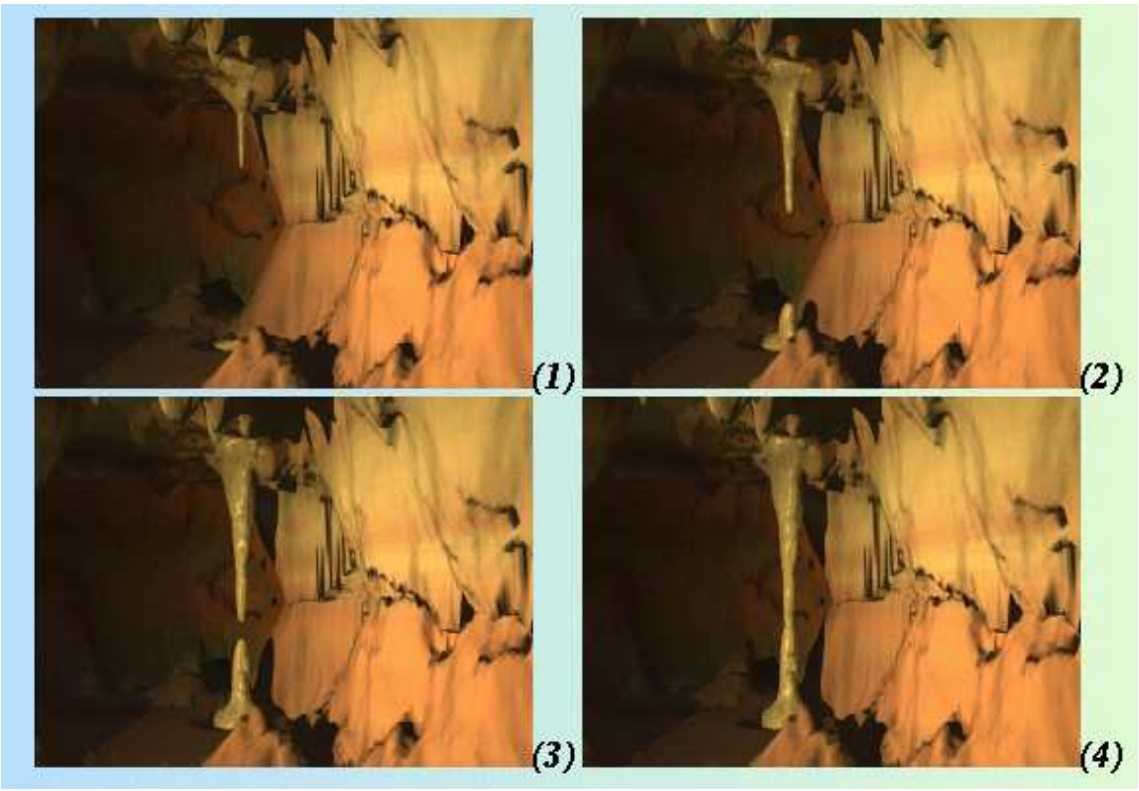

FIG. B.1 – *Caverna Magica : formation d'une colonne dans une grotte*

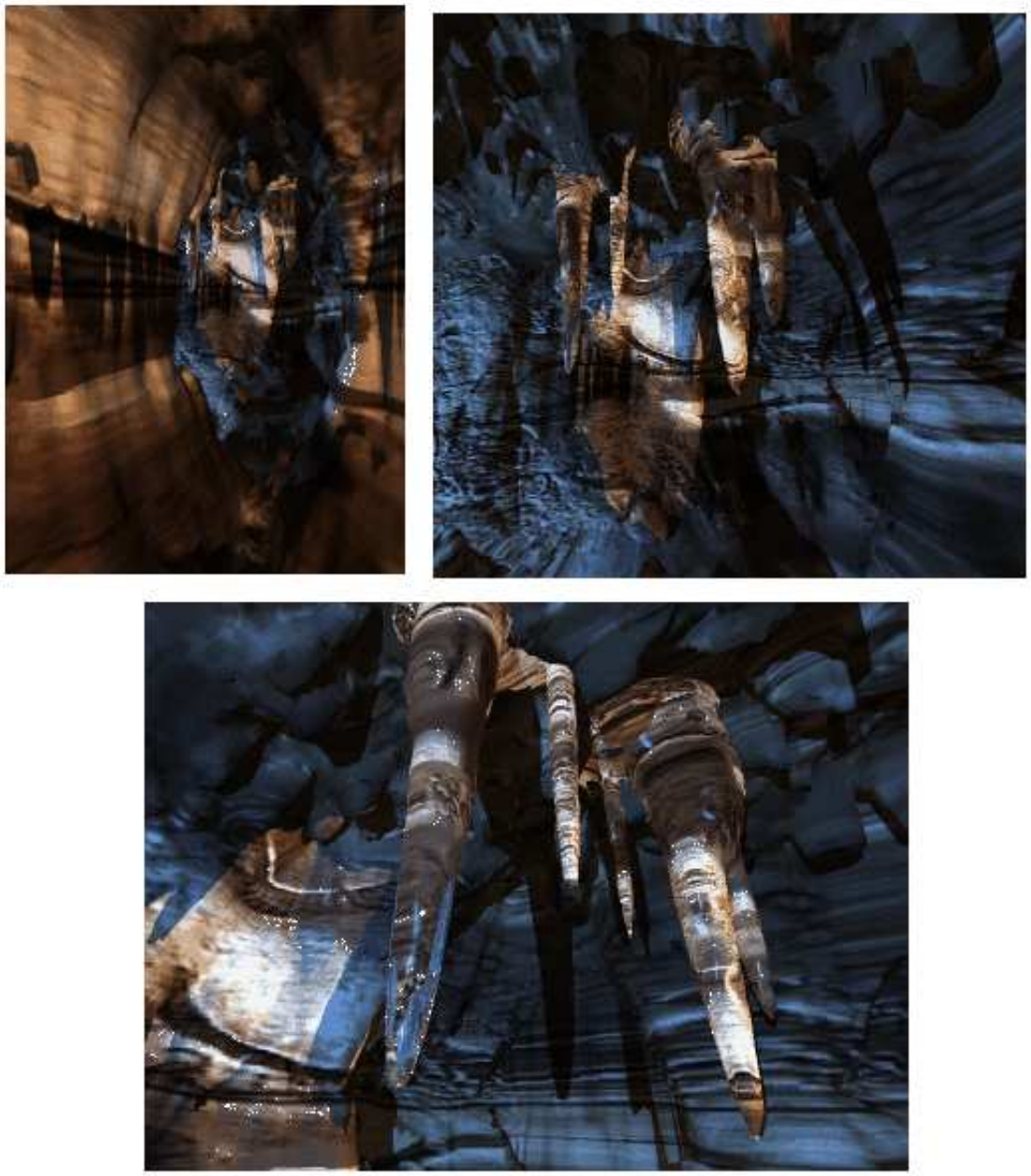

FIG. B.2 – *Caverna Magica : quelques images extraites du film*

B.2 Quelques illustrations de modélisation par fusion de formes

Dans un but démonstratif, nous avons réalisé une compilation de petites séquences vidéo illustrant la modélisation par fusion. Chaque séquence montre des effets ou des utilisations particulières des outils de déformation et du modèle que nous avons présentés.

Les planches couleurs incluses dans cette thèse correspondent à des extraits de ces séquences.

Il est possible de voir ces séquences au format MPEG sur le Web à l'adresse :

`http://www-rocq.inria.fr/syntim/research/decaudin/deform-fra.html`

La vidéo contient les manipulations et effets suivants :

- fusion d'une sphère et d'un tétraèdre (figure 2.3),
- fusion de deux bulles de couleur (figure 2.4),
- outils de déformations :
  - boîte déformée par un outil sphérique - déformations de type "bosse" et de type "creux" (figures 3.10),
  - tore déformé par un outil ellipsoïdal (figure 3.11),
  - influence de la position du centre de l'outil de fusion (figure 3.7),
  - déformation d'une ellipsoïde pour créer un modèle de chat (figure 3.12),
- modèle dédié de type fusion/flexion :
  - manipulations basiques (figures 5.4 et 5.5),
  - bras articulé et texturé (figure 4.5),
  - modèle de tigre articulé (figure 4.6),

B.3 Rendez-vous

Rendez-vous est un petit dessin animé (1mn12). Il nous a permis de tester notre algorithme de rendu style dessin animé. Nous avons déjà parlé de sa réalisation dans le chapitre 7 (section 7.3). Rappelons juste que le temps de calcul d'une image est de l'ordre de 6 minutes sur station INDY.

La figure 7.6 présente quelques images extraites du film. Il est aussi possible de voir des extraits au format MPEG sur le Web à l'adresse :

`http://www-rocq.inria.fr/syntim/research/decaudin/cartoon-fra.html`

- **Contributions : Rendez-vous** (1mn12)

Logiciel de rendu style "dessin animé", modèles, animations 3D : *Philippe Decaudin*

Storyboard : *Fabrice Neyret*

Animation des vagues : *Xavier Provot*

Montage et illustration musicale : *Christion Blonz, Arghyro Paouri*

Production : *INRIA-France, 1996*

- *Rendez-vous* a été sélectionné et présenté aux manifestations suivantes :

  - EUROGRAPHICS'96, Poitiers, France, 26-30 août 1996 ;
  - F.A.U.S.T. 96 (Forum des Arts de l'Univers Scientifique et Technologique), Toulouse, France, 18-22 octobre 1996.

Bibliographie

[Bad82] N. I. Badler. – Human body models and animation. *IEEE Computer Graphics and Applications*, vol. 2(9), novembre 1982, pp. 6–7.

[Bal96] S. Balaven. – Fusion d'objets quelconques. *Mémoire de stage INRIA - Université de Sophia Antipolis*, septembre 1996.

[Bar84] A. H. Barr. – Global and local deformations of solid primitives. *Computer Graphics (SIGGRAPH '84 Proceedings)*, vol. 18(3), juillet 1984, pp. 21–30.

[BB91] P. Borrel et D. Bechmann. – Deformation of n-dimensional objects. *Symposium on Solid Modeling Foundations and CAD/CAM Applications*, juin 1991, pp. 351–369.

[BCH+95] R. Boulic, T. Capin, Z. Huang, P. Kalra, B. Lintermann, N. Magnenat-Thalmann, L. Moccozet, T. Molet, I. Pandzic, K. Saar, A. Schmitt, J. Shen et D. Thalmann. – The HUMANOID environment for interactive animation of multiple deformable human characters. *Computer Graphics Forum (EUROGRAPHICS'95 proceedings)*, vol. 14, n° 3, août 1995, pp. 337–348.

[Bec94] D. Bechmann. – Space deformation models survey. *Computer and Graphics*, vol. 18(4), 1994, pp. 571–586.

[Bee86] E. Beeker. – Smoothing of shapes designed with free-form surfaces. *Computer Aided Design*, vol. 18(4), 1986, pp. 224–232.

[BHTT94] R. Boulic, Z. Huang, N. Magnenat Thalmann et D. Thalmann. – Goal-oriented design and correction of articulated figure motion with the TRACK system. *Computers and Graphics*, vol. 18, n° 4, 1994, pp. 443–452.

[BHW94] D. Breen, D. House et M. Wozny. – Predicting the drap of woven cloth using interacting particles. *Computer Graphics (SIGGRAPH'94 proceedings)*, vol. 25(4), juillet 1994, pp. 365–372.

[BL95] J. Bill et S. Lodha. – Sculpting polygonal models using virtual tools. *Proc. Graphics Interface'95*, mai 1995, pp. 272–279.

[Bli82] J. Blinn. – A generalization of algebraic surface drawing. *ACM Transactions on Graphics*, vol. 1(3), juillet 1982, pp. 235–256.

[BPW93] N. I. Badler, C. B. Phillips et B. L. Webber. – *Simulating Humans: Computer Graphics Animation and Control.* – New York, Oxford University Press, 1993.

[BW90] J. Bloomenthal et B. Wyvill. – Interactive techniques for implicit modeling. *Computer Graphics (SIGGRAPH'92 proceedings)*, vol. 24(2), juillet 1990, pp. 109–116.

[CC78] E. Catmull et J. Clark. – Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, vol. 10, septembre 1978, pp. 350–355.

[CHP89] J. E. Chadwick, D. R. Haumann et R. E. Parent. – Layered construction for deformable animated characters. *Computer Graphics (SIGGRAPH '89 Proceedings)*, vol. 23, n° 3, juillet 1989, pp. 243–252.

[CJ91] S. Coquillart et P. Jancène. – Animated Free-Form Deformation: an interactive animation technique. *Computer Graphics (SIGGRAPH'91 proceedings)*, vol. 25(4), juillet 1991, pp. 23–26.

[Cob84] B. S. Cobb. – *Design of Sculptured Surfaces Using the B-Spline Representation.* – Thèse de PhD, University of Utah, juin 1984.

[Coq87] S. Coquillart. – A Control-Point-Based Sweeping Technique. *IEEE Computer Graphics and Applications*, vol. 7, n° 11, novembre 1987, pp. 36–45.

[Coq90] S. Coquillart. – EFFD: a sculpturing tool for 3D geometric modeling. *Computer Graphics (SIGGRAPH'90 proceedings)*, vol. 24(4), août 1990, pp. 187–196.

[CZ92] D. T. Chen et D. Zeltzer. – Pump it up: Computer animation of a biomechanically based model of muscle using the finite element method. *Computer Graphics (SIGGRAPH '92 Proceedings)*, vol. 26(2), juillet 1992, pp. 89–98.

[Dec93] Ph. Decaudin. – Composition d'objets tridimensionnels. *Mémoire de DEA - Université de Technologie de Compiègne*, septembre 1993.

[Dec95] Ph. Decaudin. – Geometric deformation by merging a 3D object with a simple shape (version courte). *Siggraph'95 - Technical Sketches - Los Angeles, USA*, août 1995.

[Dec96a] Ph. Decaudin. – Geometric deformation by merging a 3D object with a simple shape. *Proceedings of Graphics Interface'96, Toronto, Canada*, mai 1996, pp. 55–60. – `http://www-rocq.inria.fr/syntim/textes/DeformMerging-fra.html`.

[Dec96b] Ph. Decaudin. – *Rendu de scènes 3D imitant le style dessin animé.* – Rapport de recherche n° 2919, INRIA, juin 1996. `http://www-rocq.inria.fr/syntim/textes/RR-2919-fra.html`.

[DG94] Ph. Decaudin et A. Gagalowicz. – Fusion of 3D shapes. *Fifth Eurographics Workshop on Animation and Simulation - Oslo, Norvège*, septembre 1994. – `http://www-rocq.inria.fr/syntim/textes/fusion-fra.html`.

[DG96] D. DeCarlo et J. Gallier. – Topological evolution of surfaces. *Proceedings of Graphics Interface'96, Toronto, Canada*, mai 1996, pp. 194–203.

[DH93] S. Donikian et G. Hégron. – A declarative design method for 3D scene sketch modeling. *Eurographics '93*, 1993, pp. 223–236.

[Doo78] D. Doo. – A subdivision algorithm for smoothing down irregularely shaped polyhedrons. *Proceedings on Interactive Techniques in Computer Aided Design*, 1978, pp. 157–165.

[DS88] W.-H. Du et F. Schmitt. – New results for the smooth connection between tensor product bezier patches. *New Trends in Computer Graphics (Proceedings of CG International '88)*, 1988, pp. 351–363.

[DTG95] M. Desbrun, N. Tsingos et M-P. Gascuel. – Adaptative sampling of implicit surfaces for interactive modeling and animation. *First Eurographics Workshop on Implicit Surfaces*, avril 1995.

[Duc83] B. Duc. – *L'Art de la BD, Tome 2: La Technique du Dessin.* – Editions Glénat, 1983.

[DWS93] H. Delingette, Y. Watanabe et Y. Suenaga. – Simplex based animation. *Models and Techniques in Computer Animation*, 1993, pp. 13–28.

[Els90] M. Elson. – Displacement animation: Development and application. *Course Notes #10 SIGGRAPH '90*, août 1990, pp. 14–34.

[Far86] G. Farin. – Triangular Bernstein-Bezier patches. *Computer Aided Geometric Design*, vol. 3, 1986, pp. 83–127.

[Far89] G. Farin. – *Curves and Surfaces for Computed Aided Geometric Design.* – Academic Press, 1989.

[FB88] D. R. Forsey et R. H. Bartels. – Hierarchical B-spline refinement. *Computer Graphics (SIGGRAPH '88 Proceedings)*, août 1988, pp. 205–212.

[FDFH90] J. D. Foley, A.van Dam, S. K. Feiner et J. F. Hughes. – *Computer Graphics: Principles and Practices (2e edition).* – Addison Wesley, 1990.

[For91] D. R. Forsey. – A surface model for skeleton-based character animation. *Eurographics Workshop on Animation and Simulation - Vienne, Autriche*, septembre 1991, pp. 55–73.

[Gas89] M-P. Gascuel. – Welding and pinching spline surfaces: new methods for interactive creation of complex objects and automatic fleshing of skeletons. *Proc. Graphics Interface'89*, juin 1989, pp. 20–27.

[Gas90] M-P. Gascuel. – *Déformations de surfaces complexes: Techniques de haut niveau pour la modélisation et l'animation.* – Thèse de doctorat, Université de Paris-Sud, Centre d'Orsay, octobre 1990.

[Gas93] M-P. Gascuel. – An implicit formulation for precise contact modeling between flexible solids. *Computer Graphics (SIGGRAPH'93 proceedings)*, août 1993, pp. 313–320.

[GH96] G. Giralt et G. Hirzinger. – *Robotics research.* – Springer, 1996.

[GMTT89] J-P. Gourret, N. Magnenat-Thalmann et D. Thalman. – Simulation of object and human skin deformations in a grasping task. *Computer Graphics (SIGGRAPH'89 proceedings)*, vol. 23(3), juillet 1989, pp. 21–29.

[GN94] J-D. Gascuel et J. Newmarch. – *A Tentative User and Reference Manual for TclMotif 1.0.* – non publié, février 1994.

[GP89] J. Griessmair et W. Purgathofer. – Deformation of Solids with Trivariate B-Splines. *EUROGRAPHICS'89*, 1989, pp. 137–148.

[Gra93] G. L. Graves. – The magic of metaballs. *Computer Graphics World*, mai 1993.

[GW95] A. Guy et B. Wyvill. – Controlled blending for implicit surfaces using a graph. *First Eurographics Workshop on Implicit Surfaces*, avril 1995.

[HCV56] D. Hilbert et S. Cohn-Vossen. – *Geometry and the Imagination.* – Chelsea Publishing Compagny, 1956.

[HH87] C. Hoffmann et J. Hopcroft. – The potential method for blending surfaces and corners. – *Geometric Modeling: Algorithms and New Trends*, éd. par G. Farin, pp. 347–365. – SIAM, Philadelphia, 1987.

[HHK92] W. M. Hsu, J. F. Hughes et H. Kaufman. – Direct manipulation of free-form deformations. *Computer Graphics (SIGGRAPH'92 proceedings)*, vol. 26(2), juillet 1992, pp. 177–184.

[Hug92] J. F. Hughes. – Scheduled Fourier Volume Morphing. *Proc. Computer Graphics, SIGGRAPH'92*, vol. 26, n° 2, juillet 1992, pp. 43–46.

[KCP92] J. R. Kent, W. E. Carlson et R. E. Parent. – Shape transformation for polyhedral objects. *Computer Graphics (SIGGRAPH'92 proceedings)*, vol. 26(2), juillet 1992, pp. 47–54.

[Klo86] F. Klok. – Two Moving Coordinate Frames for Sweeping along a 3d Trajectory. *Computer Aided Geometric Design*, no3, 1986, pp. 217–229.

[Kol91] C. E. Kolb. – *Rayshade User's Guide and Reference Manual.* – non publié, 1991.

[KR91] A. Kaul et J. Rossignac. – Solid-interpolation deformations: Construction and animation of PIPs. *Eurographics'91 proceedings*, septembre 1991, pp. 493–505.

[Laz95] F. Lazarus. – *Courbes, Cylindres et Métamorphoses.* – Thèse de doctorat, Université de Paris VII, décembre 1995.

[LCJ93] F. Lazarus, S. Coquillart et P. Jancène. – Interactive axial deformations. – *Modeling in Computer Graphics*, éd. par Springer-Verlag, pp. 241–254. – genèvre, juin 1993.

[Loo94] C. Loop. – Smooth spline surfaces over irregular meshes. *Computer Graphics (SIGGRAPH '94 Proceedings)*, juillet 1994, pp. 303–310.

[Mar82] D. Marr. – *Vision.* – San Francisco, Freeman, 1982.

[Mei96] B. J. Meier. – Painterly rendering for animation. *Computer Graphics (SIGGRAPH '96 Proceedings)*, août 1996, pp. 477–484.

[MTT91] N. Magnetat-Thalmann et D. Thalmann. – Human shape design and deformations. *Course Notes #20 SIGGRAPH '91 - Advanced Techniques in Human Modeling, Animation, and Rendering*, août 1991.

[NDW95] J. Neider, T. Davis et M. Woo. – *OpenGL Programming Guide (Release 1).* – Addison Wesley, 1995.

[New75] M. Newell. – *The Utilization of Procedure Models in Digital Image Synthesis.* – Ph.d. thesis, University of Utah, 1975.

[NHTa85] T. Nishita, M. Hirai, T.Kawai et al. – Object modeling by distribution function and a method of image generation. *Trans. IECE - Journal of Papers given at the Electronics Communications Conference (in Japanese)*, vol. J68-D(4), juillet 1985, pp. 718–725.

[OM93] A. Opalach et S. Maddock. – Implicit surfaces: Appearence, blending and consistency. *4th Eurographics Workshop on Animation and Simulation*, 1993.

[Ous93] J. K. Ousterhout. – *Tcl and the Tk Toolkit.* – Addison-Wesley, 1993.

[Ped95] H. Kohling Pedersen. – Decorating implicit surfaces. *Computer Graphics (SIGGRAPH '95 proceedings)*, août 1995, pp. 291–300.

[RK94] J. Rossignac et A. Kaul. – AGRELs and BIBs: Metamorphosis as a bézier curve in the space of polyhedra. *Eurographics'94 proceedings*, septembre 1994, pp. 179–184.

[RSC87] W. T. Reeves, D. H. Salesin et R. L. Cook. – Rendering antialiased shadows with depth maps. *Computer Graphics (SIGGRAPH '87 Proceedings)*, vol. 21(4), juillet 1987, pp. 283–291.

[SABS94] M. P. Salisbury, S. E. Anderson, R. Barzel et D. H. Salesin. – Interactive pen-and-ink illustration. *Computer Graphics (SIGGRAPH '94 Proceedings)*, vol. 28, juillet 1994, pp. 101–108.

[SBD86] F. Schmitt, B. A. Barsky et W.-H. Du. – An adaptive subdivision method for surface-fitting from sampled data. *Computer Graphics (SIGGRAPH '86 Proceedings)*, vol. 20, août 1986, pp. 179–188.

[SG92] T.W. Sederberg et E. Greenwood. – A Physically Based Approach to 2D Shape Blending. *Proc. Computer Graphics, SIGGRAPH'92*, vol. 26, juillet 1992, pp. 25–34.

[Shn83] B. Shneiderman. – Direct manipulation: A step beyond programming languages. *IEEE Computer*, vol. 16(8), août 1983, pp. 57–69.

[Smi84] A. R. Smith. – Plants, fractals and formal languages. *Computer Graphics (SIGGRAPH '84 Proceedings)*, vol. 18, juillet 1984, pp. 1–10.

[SP86] T.W. Sederberg et S.R. Parry. – Free-Form Deformation of solid geometric models. *Computer Graphics (SIGGRAPH'86 proceedings)*, vol. 20(4), août 1986, pp. 151–160.

[SS96] J-P. Smets-Solanes. – Vector field based texture mapping of animated implicit objects. *Computer Graphics Forum (Eurographics'96)*, août 1996, pp. 289–300.

[ST90] T. Saito et T. Takahashi. – Comprehensible rendering of 3-D shapes. *Computer Graphics (SIGGRAPH '90 Proceedings)*, vol. 24(4), août 1990, pp. 197–206.

[ST92] R. Szeliski et D. Tonnesen. – Surface modeling with oriented particle systems. *Computer Graphics (SIGGRAPH'92 proceedings)*, vol. 26(2), juillet 1992, pp. 185–194.

[ST95] J. Shen et D. Thalmann. – Interactive shape design using metaballs and splines. *First Eurographics Workshop on Implicit Surfaces*, avril 1995.

[SW91] Y. Suenaga et Y. Watanabe. – A method for synchronized acquisition of cylindrical rang and color data. *IEICE Transactions*, vol. E 74(10), octobre 1991, pp. 3407–3416.

[Tea93] POV-Ray Team. – *Persistence of Vision Ray Tracer, User's Documentation.* – non publié, 1993.

[TF88] D. Terzopoulos et K. Fleisher. – Deformable models. *The Visual Computer*, vol. 4(6), 1988, pp. 306–331.

[TM91] D. Terzopoulos et D. Metaxas. – Dynamic 3-d models with local and global deformations: deformable super quadrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-13(7), juillet 1991, pp. 703–714.

[TPBF87] D. Terzopoulos, J. Platt, A. Barr et K. Fleischer. – Elastically deformable models. *Computer Graphics (SIGGRAPH'87 proceedings)*, juillet 1987, pp. 205–214.

[TW88] D. Terzopoulos et A. Witkin. – Physically based models with rigid and deformable components. *IEEE Computer Graphics and Applications*, vol. 8(6), novembre 1988, pp. 41–51.

[TWK87] D. Terzopoulos, A. Witkin et M. Kass. – Symmetry-seeking models for 3D object reconstruction. *International Journal of Computer Vision*, vol. 1, 1987, pp. 211–221.

[WBB+90] B. Wyvill, J. Bloomenthal, T. Beier, J. Blinn, A. Rockwood et G. Wyvill. – Modeling and animating with implicit surfaces. *Siggraph Course Notes*, vol. 23, 1990.

[Wer95] J. Wernecke. – *The Inventor Mentor - Programming Object-Oriented 3D Graphics with Open Inventor (Release 2).* – Addison Wesley, 1995.

[Wil78] L. Williams. – Casting curved shadows on curved surfaces. *Computer Graphics (SIGGRAPH '78 Proceedings)*, vol. 12(3), août 1978, pp. 270–274.

[WMW86a] B. Wyvill, C. McPheeters et G. Wyvill. – Animating soft objects. *The Visual Computer*, vol. 2, 1986, pp. 235–242.

[WMW86b] G. Wyvill, C. McPheeters et B. Wyvill. – Data structure for soft objects. *The Visual Computer*, vol. 2, 1986, pp. 227–234.

[WS94] G. Winkenbach et D. H. Salesin. – Computer-generated pen-and-ink illustration. *Computer Graphics (SIGGRAPH '94 Proceedings)*, vol. 28, juillet 1994, pp. 101–108.

[WS96] G. Winkenbach et D. H. Salesin. – Rendering parametric surfaces in pen and ink. *Computer Graphics (SIGGRAPH'96 proceedings)*, août 1996, pp. 469–476.

[WW92] W. Welch et A. Witkin. – Variational Surface Modeling. *Proc. Computer Graphics, SIGGRAPH'92*, vol. 26, n° 2, juillet 1992, pp. 157–166.

Thèse de doctorat en sciences de l'Université de Technologie de Compiègne

Modélisation par Fusion de Formes 3D pour la Synthèse d'Images, Rendu de Scènes 3D imitant le Style «Dessin Animé»

Philippe Decaudin

Résumé : Nous proposons, dans une première partie, de nouveaux outils de modélisation d'objets tridimentionnels pour la synthèse d'images. Ils permettent de modeler interactivement des formes d'aspect lisse telles que des formes organiques (animaux, corps humains) et facilitent leur animation et leur texturation. Un objet de forme complexe est créé en appliquant une succession de déformations de type *fusion* ou *flexion* à un objet simple. L'outil de fusion permet de modeler l'objet en le *fusionnant* avec une forme 3D simple (sphère, ellipsoïde, ...) ; l'objet est déformé de façon à englober la forme simple. L'outil de flexion est utilisé pour créer des articulations qui permettront d'animer l'objet.

Dans la deuxième partie, nous proposons un algorithme de rendu *non-photoréaliste*. Il génère des images imitant le style «dessin animé» traditionnel (ou «bande dessinée») à partir de la description tridimensionnelle d'une scène fixe ou animée. Pour ce faire, l'algorithme fait appel à des techniques qui permettent de dessiner les contours des objets (profils et arêtes sont dessinés en traits noirs), de colorer uniformément les surfaces intérieures à ces contours, et de faire apparaître sur les objets les ombres propres et les ombres portées dues aux sources de lumière éclairant la scène.

Mots-Clés : Synthèse d'Images, Modélisation, Déformation, Techniques interactives, Animation, Rendu non-photoréaliste, Dessin animé.

Abstract : In the main section, we introduce new tools for modeling three-dimensionnal objects for computer graphics. They allow interactive modeling of smooth shapes such as organic-looking shapes (animals, human bodies) and help animating and texturing them. A complex object is created by applying a succession of *fusion* and *flexion* deformations to a simple object. The fusion tool allows deformation of the shape of the object by merging it with a simple 3D-shape (sphere, ellipsoid, ...); the object is deformed so that it embeds the simple shape. The twist tool allows creation of articulations which can be used to animate the deformable object.

In a second section, we introduce a *non-photorealistic* rendering algorithm. It produces images having the appearance of a traditional cartoon from a 3D description of the scene (a static or an animated scene). The 3D scene is rendered with techniques allowing to outline the profiles and edges of objects, to color uniformly the patches, and to render shadows (self-shadows and projected-shadows) due to light sources.

Keywords : Computer graphics, Modeling, Deformation, Interactive techniques, Animation, Non-photorealistic rendering, Cartoon.