Un modeleur interactif d'objets définis par des surfaces implicites

Nicolas Tsingos
Marie-Paule Gascuel

Université Joseph Fourier
iMAGIS[†] / IMAG - INRIA
BP 53
38041 Grenoble cedex 09, France
télécopie: +33 76 44 66 75
e-mail: Nicolas.Tsingos@imag.fr
Marie-Paule.Gascuel@imag.fr

RÉSUMÉ: Cet article présente un outil de modélisation interactive d'objets définis par des surfaces implicites engendrées par des squelettes. Basé sur une nouvelle technique de discrétisation adaptative, il offre deux modes complémentaires de visualisation temps réel, et permet un contrôle simple des objets via la manipulation interactive de leur squelettes et des fonctions potentiel associées.

ABSTRACT: This article presents a modeling tool for interactive design of objects defined by implicit surfaces. Based on a new sampling technique, it offers to complementary real-time visualizing modes and allows the user to control the defined objects in a very simple way, by interactively manipulating their skeletons and the associated field functions.

MOTS-CLÉS: modélisation, surfaces implicites, visualisation, interactivité

KEYWORDS: modelisation, implicit surfaces, visualization, interactivity

1 Introduction

Les surfaces implicites font l'objet d'un intérêt croissant dans la communauté graphique internationale depuis quelques années. En effet, elles permettent de représenter très facilement des objets lisses de forme libre, y compris lorsque ces objets comportent des embranchements, ou qu'ils sont de topologie arbitraire (objets munis de "poignées"). De plus, elles semblent particulièrement prometteuses pour animer des objets déformables, pouvant changer de topologie. On les retrouve dans des domaines aussi variés que l'animation descriptive [WMG86, OSM93], l'animation par modèles physiques [PW89, TM91, Gas93, DG94], ou la modélisation de métamorphoses [Wyv93].

[†] iMAGIS est un projet commun entre le CNRS, l'INRIA, l'Institut National Polytechnique de Grenoble et l'Université Joseph Fourier

Malheureusement, la modélisation interactive par surfaces implicites semble encore absente de la plupart des logiciels du commerce. L'une des principales raisons est la difficulté de visualisation temps réel. En effet, si un rendu de qualité comme le lancer de rayon est facile à étendre aux surfaces implicites (une intersection rayon-surface peut être calculée par dichotomie), il n'en est pas de même pour les procédés de visualisation interactive standard: en effet, on ne dispose pas à priori de points sur une surface implicite, ni même d'une équation paramétrique qui permettrait d'en calculer. Les techniques d'échantillonnage existantes, spécifiques aux surfaces implicites et procédant par suivi de la surface, ont des performances de l'ordre de la minute, ce qui semble inacceptable dans le cas de l'édition interactive d'un objet. Ces problèmes ont limité jusqu'à présent l'emploi des surfaces implicites, aussi bien en modélisation qu'en animation.

Cet article présente un logiciel interactif pour la modélisation et l'édition d'objets définis par des surfaces implicites. De manière à faciliter le contrôle de la forme des objets, nous avons choisi des surfaces implicites définies comme des iso-surfaces autour de "squelettes" aux formes géométriques simples. Outre un ensemble d'outils de manipulation interactive des squelettes et des fonctions potentiel qu'ils engendrent, nous proposons une nouvelle méthode d'échantillonnage adaptatif des surfaces implicites, qui reste efficace même en cas de changements de topologie. Il en découle plusieurs modes de visualisation interactive, qui permettent de donner une très bonne idée de la forme de l'objet après toute déformation.

La Section 2 rappelle les approches traditionnelles qui permettent de définir des objets "implicites" et de les visualiser. Nous abordons dans la Section 3 les choix de conception de notre modeleur. La Section 4 décrit les modes de visualisation interactive offerts, ansi que l'algorithme d'échantillonage adaptatif sur lequel ils sont basés. Enfin, nous décrivons dans la Section 5 l'implémentation de l'outil et l'interface offerte, avant de présenter nos résultats et de conclure.

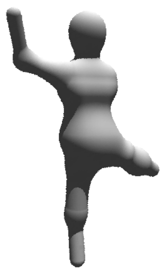
2 Les surfaces implicites

2.1 Définitions

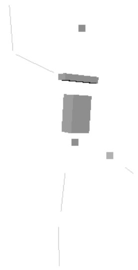
Une surface implicite, également appelée "iso-surface", peut être définie comme l'ensemble des points de l'espace P(x,y,z) verifiant f(P)=c où c est une constante appelée "isovaleur".

Une première possibilité consiste à définir une surface implicite directement par la donnée de son équation analytique [Bar81], puis de la déformer grâce à des transformations matricielles [Bar84]. Cependant, afin de faciliter le contrôle de la forme modélisée, de nombreux auteurs se sont intéressés à des surfaces engendrées par des "squelettes", qui peuvent être soit des points [Bli82, WMW86], soit d'autres primitives géométriques simples [BW90, BS91]. Nous nous limitons dans le cadre de cet article à ce second type de surfaces, qui semble offrir un contrôle plus intuitif de la forme.

Un *squelette* est une primitive géométrique pour laquelle on peut définir une fonction de distance. Chaque squelette s_i est muni d'une fonction f_i, appelée "*fonction potentiel*", dont la valeur décroit lorsqu'on s'éloigne du squelette. La surface implicite S



(a) Venus “implicite”...



(b) ...et ses squelettes

FIG. 1 - *Exemple d’objet “implicite” et squelettes utilisés*

est alors définie par :

$$S = \left\{ P \in \mathbb{R}^3 \mid f(P) = \sum_{s_i} f_i(r_i) = c \right\}$$

où r_i est la distance de P à s_i et f_i est une fonction décroissante. Un exemple d’objet ainsi défini est donné sur la Figure 1.

2.2 Visualisation d’une surface implicite

Les surfaces implicites peuvent être rendues avec une grande précision par lancer de rayon. Il n’est alors pas nécessaire de passer par une discrétisation : les intersections rayon/surfaces sont calculées par dichotomie (on sait en effet qu’un point du rayon est à l’intérieur de l’objet si et seulement si $f(P) >= c$). Leur principal inconvénient réside dans la difficulté de les visualiser de manière interactive. Pour cela il est nécessaire de disposer de points d’échantillonnage à la surface des objets. La plupart des méthodes existantes construisent une représentation en polygones afin d’exploiter la vitesse des stations graphiques spécialisées. Elles sont généralement basées sur une discrétisation de l’espace en cubes. Ainsi, on commence par trouver un cube initial qui intersecte la surface, puis on procède à un suivi de la surface en examinant les cubes voisins qui l’intersectent, des polygones étant construits sur chacun de ces cubes [WMW86, LC87, Blo88, ZJ91, WG92, NB93]. Toutefois, ces techniques ne sont pas suffisamment rapides pour assurer un recalcul interactif de la polygonisation après chaque modification de l’objet. Bloomenthal et Wyvill proposent en 1990 de remplacer le calcul d’une polygonisation par celui beaucoup moins lourd d’un nuage de points sans topologie en facettes [BW90]. Dans cette méthode, des points (ou graines) supposés suffisamment proches de la surface migrent vers celle-ci en suivant la direction du gradient de la fonction potentiel. Aucune indication n’est toutefois donnée pour déterminer de manière satisfaisante les positions initiales des graines, qui peuvent ne pas être triviales lorsque la géométrie de l’objet est quelconque (en particulier si il comporte des embranchements ou des trous). De plus, le nombre et la répartition des graines nécessaires seraient amené à changer lors d’une phase de modélisation interactive (rupture en plusieurs composantes connexes lors du déplacement d'un squelette ; fusion de deux objets), ce qui rend cette méthode inutilisable telle quelle. Witkin et Heckbert proposent en 1994 une méthode d'échantillonnage basée sur un système de particules [WH94]. Lorsque le système atteint un état d'équilibre, l'échantillonnage obtenu est uniforme. Ils introduisent, en particulier, des particules de "taille" adaptative pouvant se subdiviser (la taille d'une particule correspond au rayon de non-interpénétration autour de celle-ci). Après chaque déformation de la surface, de "grosses" particules sont utilisées pour échantillonner rapidement les zones sous-échantillonnées. Lorsqu'elles ont atteint un état d'équilibre, elles sont subdivisées jusqu'à atteindre la précision souhaitée. Un processus d'optimisation sous contraintes est en outre utilisé pour garantir que les particules restent sur la surface. Néanmoins, ajouter ou supprimer des squelettes reste impossible dans l'implantation courante de ce système, le nombre et la nature des degrés de libertés devant être connu à l'avance, ce qui limite les applications en modélisation interactive. De plus, les auteurs ne proposent pas de méthode de visualisation opaque des objets, ce qui peut être gênant dans le cas de formes complexes.

3 Conception d'un modeleur

Un outil de modélisation doit avant tout être simple et intuitif de manière à pouvoir être utilisable par un graphiste. Dans ce cadre, les surfaces implicites engendrées par des squelettes, décrites à la section 2.1, semblent les mieux adaptées, les squelettes donnant une bonne idée de la forme et permettant une manipulation interactive immédiate.

Il faut également que les fonctions potentiel associées à ces squelettes ne soit pas trop couteuses à évaluer et que leurs paramètres de contrôle soient simples. En effet, ces fonction jouent un grand rôle sur la forme dans les zones où les influences de plusieurs squelettes se mélangent. Nous avons choisi d'utiliser des fonctions paramétrées par trois valeurs (Figure 2) :

- un *rayon d'influence* R (au delà duquel le potentiel s'annule) permettant d'obtenir un contrôle local de la forme de l'objet.
- une *épaisseur* e correspondant à l'épaisseur de surface autour du squelette lorsque ce dernier est isolé.
- une *raideur* k définie comme la pente de la fonction potentiel en e. Cette raideur contrôle le mélange des potentiels.

On peut ainsi stocker de manière compacte les objets modélisés (e,k,R, positions et type des squelettes). On peut également calculer la normale aux objets de manière exacte, grâce au calcul analytique du gradient de la fonction potentiel.

En résumé, les principales fonctionnalités du modeleur proposé sont :

- Visualisation de la scène dans une ou plusieurs fenêtres 3D.
- Ajout et Suppression d'un squelette. Copier/Coller.
- Manipulation interactive des squelettes/objets.

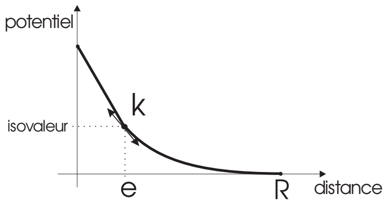


FIG. 2 - *Fonction potentiel et ses paramètres de contrôle*

– Edition interactive et graphique des fonctions potentiel de chaque squelette dans une fenêtre 2D.

4 Visualisation interactive

Il ne s'agit pas ici d'obtenir un rendu de qualité de l'objet mais de définir une méthode de visualisation interactive capable de donner à l'utilisateur une bonne idée de la forme qu'il est en train de modéliser. Nous proposons une nouvelle technique inspirée de l'idée de la migration de graines, évoquée au paragraphe 2.2.

4.1 Une discrétisation efficace et adaptative

L'algorithme mis au point peut se décomposer en deux phases : une phase d'initialisation et une phase de migration.

Initialisation des graines et choix des directions de migration

Nous déterminons une boite englobant chaque squelette, augmentée de e où e est le paramètre d'épaisseur de la fonction potentiel associée à ce squelette (Figure 2a). Nous discrétisons alors uniformément les faces de cette boite en fonction d'un pas de discrétisation donné par l'utilisateur. En projetant ces points de discrétisation (graines) sur le squelette, nous obtenons les directions de recherche souhaitées (Figure 2b). Afin que les positions des graines ne soient pas recalculées en cas de simples translations ou rotations, elles sont exprimées dans le repère local à leur squelette. La position d'une graine dans ce repère est alors donnée par son axe de déplacement et son abscisse sur cet axe. Ayant déterminé les axes de déplacement de chaque graine, celles ci sont positionnées à l'abscisse e sur leurs axes respectifs (Figure 2c). Cette phase d'initialisation est effectuée pour chacun des squelettes de l'objet.

La phase de migration

Pour trouver une position valide d'une graine sur la surface de l'objet complet, on effectue tout d'abord des “pas” de recherche, uniformes et dépendant de e, de manière à obtenir un point intérieur à l'objet et un autre extérieur. Puis, pour trouver l'intersection exacte avec la surface, on procède par simple dichotomie entre ces deux points. On obtient ainsi la nouvelle position de la graine (Figure 4). Dans le cas où l'objet a été déformé, on procède de manière identique en partant de la dernière position valide de

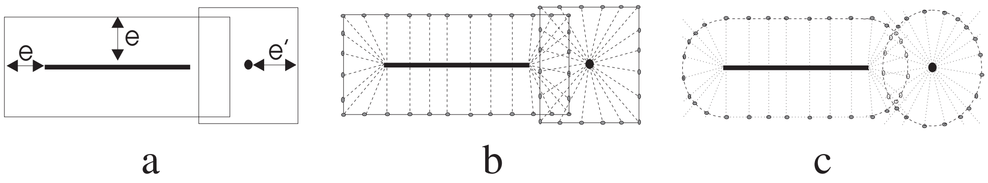


FIG. 3 - *Initialisation des graines*

la graine.

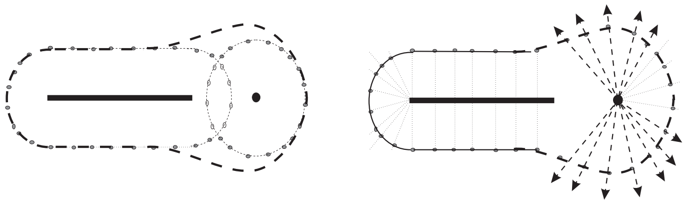


FIG. 4 - *Migration des graines*

Validation des graines

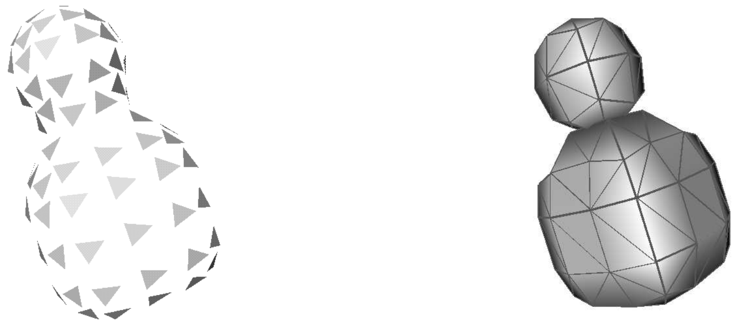
Comme on le voit sur la Figure 2c, certaines graines (celles situées entre les deux squelettes) sont inutiles pour discrétiser l'objet. Il est donc nécessaire de pouvoir les invalider afin de ne pas en tenir compte. Connaissant quel squelette a "lancé" une graine, on peut savoir, en chaque position successive de celle-ci, si le potentiel engendré par "son" squelette est prépondérant par rapport aux autres potentiels. Si c'est le cas, la graine est valide. Sinon, la graine est entrée dans une zone déja discrétisée par les graines d'un autre squelette ; elle est alors inutile et on l'invalide. Bien évidemment, une graine invalidée peut redevenir valide lors d'un changement de topologie au cours de l'édition interactive de l'objet. L'échantillonnage obtenu est ainsi adaptatif.

4.2 Modes de visualisation

Visualiser le nuage de points constitué par les graines valides serait insuffisant pour donner une bonne idée de la forme de l'objet. Nous proposons deux modes de visualisation.

Visualisation par "écailles" : Premièrement on peut centrer sur chaque graine une "écaille" (une facette triangulaire par exemple) orientée suivant la normale à la surface en ce point. Ces facettes sont visualisées en temps réel par les stations graphiques spécialisées et donnent une assez bonne idée de la forme de l'objet (Figure 5).

Visualisation polygonale par morceaux : Lorsque les objets ou les scènes modélisées deviennent très complexes, il est souhaitable d'avoir également un affichage continu et opaque des objets. Plutôt que de calculer une triangulation de Delaunay après chaque modification, ce qui serait très couteux en temps de calcul, nous affichons une polygonisation "par morceaux" des objets, un maillage polygonal étant associé aux graines lancées par chacun des squelettes. Ces maillages sont construits durant la phase d'initialisation des graines, à partir de la discrétisation calculée sur chaque boite englobante. Ainsi, ils se déforment lors du mouvement des graines, mais ne nécessitent aucun calcul supplémentaire au cours des déformations interactives de l'objet, les graines invalides comme les valides étant prises en compte. On obtient ainsi une visualisation opaque "par blocs" de l'objet. La prise en compte des graines invalides introduit des discontinuités apparentes de tangente aux jointures entre les différents blocs (Figure 5b). Néanmoins ce phénomène n'est pas génant dans le cadre de notre application puisque l'utilisateur peut facilement commuter l'affichage en mode "écailles" pour avoir une idée plus précise de la courbure de l'objet. En outre, ce phénomène est atténué dès qu'on augmente le taux d'échantillonnage (Figure 6a). Ce mode de visualisation original sans aucun calcul ajouté est particulièrement interessant. En effet, aucune autre approche, n'offre à la fois un recalcul peu couteux de l'échantillonnage après une déformation (car exploitant la cohérence temporelle) et une visualisation opaque des objets, particulièrement utile pour la création de scènes complexes, ou dans le cadre d'animation d'objets en collision.



(a) Visualisation par "écailles" (b) Visualisation par polygones

FIG. 5 - *Visualisation par "écailles" et par polygones (80 points d'échantillonnage)*

5 Implémentation et résultats

L'outil de modélisation à été implémenté dans le cadre du système *FABULE* développé au sein de l'équipe iMAGIS. Il est écrit en *Tcl* , un langage interprété (aparenté à des langages shell unix) qu'on peut facilement étendre par de nouvelles commandes. Les commandes et procédures Tcl s'appuient sur des fonctions C++ du système *FABULE* prenant en charge les opérations de bas niveau, en particulier l'algorithme de discrétisation et l'affichage. Les primitives graphiques sont gérées sous forme de "noeuds" *OpenInventor*, une base de données graphique 3D interactive. Les opérations graphiques

(a) polygones
(960 points échantillonnés)

(b) lancer de rayon

FIG. 6 - *Comparatif de la visualisation par polygones et par lancer de rayon*

de base sont donc effectuées en utilisant les fonctions offertes par cette librairie. Enfin,l'interface s'appuie sur *OSF/Motif* pour la gestion des fenêtres, boutons, etc. (Figure 7)

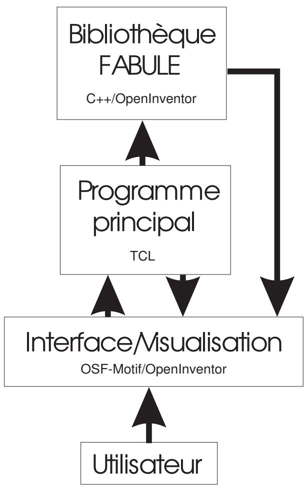
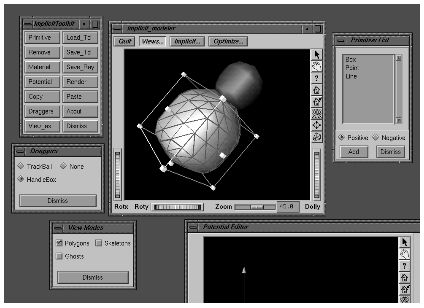
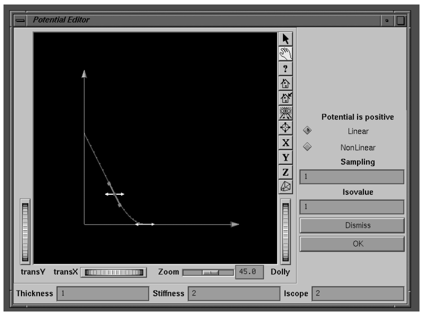


FIG. 7 - *"Architecture logicielle" du modeleur*

Celle-ci consiste principalement en une fenêtre de rendu et une "barre d'outils" permettant de visualiser et éditer l'objet en cours de modélisation (Figure 8a). Les différents modes de visualisation peuvent être combinés entre eux en particulier grâce à la gestion de matériaux semi-transparents permettant, par exemple, une visualisation simultanée de la surface et des squelettes sous-jacents (Figure 9a). D'autres fenêtres de vues 2D ou 3D peuvent être ouvertes afin de faciliter la manipulation de l'objet. Après avoir sélectionné un objet dans la (ou les) fenêtre(s) de rendu, on peut donc le

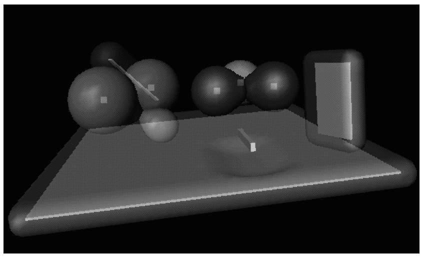


(a) Vue générale de l'interface

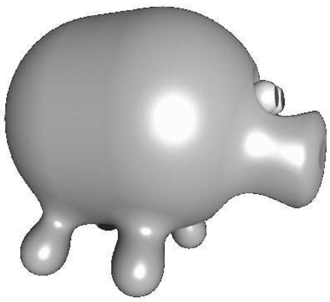


(b) Editeur de fonctions potentiel

FIG. 8 - *L'interface du modeleur*



(a) Visualisation interactive d'une scène durant la modélisation.



(b) Une tirelire implicite

FIG. 9 - *Résultats*

déplacer, modifier sa taille grâce aux manipulateurs proposés, ou bien lui appliquer une des diverses fonctions offertes par la barre d'outils. Ainsi, on pourra éditer interactivement, de manière graphique, les fonctions potentiel des différents squelettes (Figure 8b), choisir le mode de visualisation, ajouter ou supprimer des squelettes, copier/coller, changer les matériaux (couleurs, propriétés), effectuer un rendu test en raytracing et sauvegarder son travail sous différents formats. *Tcl* constitue également le format d'entrées/sorties standard du modeleur, une scène étant sauvegardée sous la forme d'un script de commandes permettant de la recréer. Ainsi on peut facilement intégrer un objet modélisé à un script plus complexe, contrôlant son animation par exemple.

Parfois, il est plus simple de construire un objet en "creusant" une forme de base à l'aide d'un outil dont la forme à été soigneusement choisie. On peut réaliser ce type d'opération en utilisant des squelettes à influence négative (leur potentiel n'est alors plus ajouté mais soustrait) comme on peut le voir sur la Figure 9a, où une barre creuse localement le sol au premier plan.

6 Conclusion et perspectives

Nous avons présenté un outil de création et de manipulation interactive d'objets définis par des surfaces implicites. L'utilisateur contrôle les "squelettes" définissant ces surfaces dans l'une ou plusieurs fenêtres de vue (déplacement, déformation d'un squelette, ajout, suppression, couper/coller), et dispose également d'une interface graphique pour modifier les potentiels qu'ils engendrent. En particulier, il dispose de l'accès direct à l'épaisseur d'enrobage d'un squelette, et, par le biais de la "raideur" de la fonction, à son mode de mélange avec les contributions des squelettes voisins.

L'interactivité de l'outil, essentielle pour le rendre convivial, est rendue possible grâce à une nouvelle méthode d'échantillonnage adaptatif des surfaces implicites. Particulièrement efficace, cette méthode s'adapte facilement aux changements de topologie qui peuvent intervenir lors de la modélisation d'un objet. Nous lui avons associé deux modes complémentaires de visualisation temps réel qui offrent une vision assez précise de la forme modélisée. En particulier, nous proposons un mode original permettant de visualiser la surface modélisée de manière opaque et sans calculs ajoutés, exploitant la cohérence temporelle. Les positions des squelettes et paramètres des fonctions implicites offrent un stockage très compact de l'objet, qui peut être utilisé pour un rendu final de qualité (lancer de rayons direct).

En conclusion, cet outil permet la création interactive d'objets complexes en combinant peu à peu des éléments plus simple (ajouts et édition successives des squelettes), tout en offrant la propriété fondamentale qu'est le contrôle local de la forme (l'influence de chaque squelette étant limitée dans l'espace).

L'un des problèmes les plus importants posé par la modélisation à l'aide de surfaces implicites est celui du "mélange indésirable". En effet lorsque l'on modélise un personnage, ses bras doivent se mélanger avec ses épaules mais avec aucune autre partie de son corps ! Des solutions à ce problème ont déjà été proposées [WW89, OSM93, GW95] et sont basées sur la définition d'un graphe de voisinage entre les squelettes d'un objet, chaque squelette ne se mélangeant qu'avec ses voisins. Une méthode s'en inspirant a été implantée dans le cadre du système *FABULE* [DTG95] et peut facilement être intégrée au sein de notre outil. Enfin, un autre moyen de contrôle des objets modélisés, en cours d'étude actuellement, est l'introduction de nouvelles fonctions potentielles, anisotropes, qui ne dépendent pas simplement de la distance au squelette. Introduites dans [KAW91], elles peuvent permettre, en particulier, d'engendrer des cônes généralisés à l'aide de squelettes simples.

Remerciements

Nous remercions toute l'equipe iMAGIS pour son soutien et plus particulièrement Jean-Dominique Gascuel, sans qui la réalisation de ce projet eut été impossible, ainsi que Mathieu Desbrun avec qui nous avons eu grand plaisir à discuter.

Références

[Bar81] Alan H. Barr. Superquadrics and angle-preserving transforms. *IEEE Computer*

Graphics and Applications, 1(1):11–23, 1981.

[Bar84] Alan H. Barr. Global and local deformations of solid primitives. *Computer Graphics*, 18(3):21–29, 1984.

[Bli82] J. Blinn. A generalization of algebraic surface drawing. *ACM Transactions on Graphics*, pages 235–256, July 1982.

[Blo88] Jules Bloomenthal. Polygonisation of implicit surfaces. *Computer Aided Geometric Design*, 5:341–355, 1988.

[BS91] Jules Bloomenthal and Ken Shoemake. Convolution surfaces. *Computer Graphics*, 25(4):251–256, July 1991. Proceedings of SIGGRAPH'91 (Las Vegas, Nevada, July 1991).

[BW90] Jules Bloomenthal and Brian Wyvill. Interactive techniques for implicit modeling. *Computer Graphics*, 24(2):109–116, March 1990.

[DG94] Mathieu Desbrun and Marie-Paule Gascuel. Highly deformable material for animation and collision processing. In *5th Eurographics Workshop on Animation and Simulation*, Oslo, Norway, September 1994.

[DTG95] Mathieu Desbrun, Nicolas Tsingos, and Marie-Paule Gascuel. Adaptive sampling of implicit surfaces for interactive modeling and animation. In *Implicit Surfaces'95*, Grenoble, April 1995.

[Gas93] Marie-Paule Gascuel. An implicit formulation for precise contact modeling between flexible solids. *Computer Graphics*, pages 313–320, August 1993. Proceedings of SIGGRAPH'93 (Anaheim, California, August 1993).

[GW95] Andrew Guy and Brian Wyvill. Controlled blending for implicit surfaces using a graph. In *1st Eurographics Workshop on Implicit Surfaces*, April 1995.

[KAW91] Zoran Kacic-Alesic and Brian Wyvill. Controlled blending of procedural implicit surfaces. In *Graphics Interface'91*, pages 236–245, Calgary, Canada, June 1991.

[LC87] William Lorensen and Harvey Cline. Marching cubes: a high resolution 3d surface construction algorithm. *Computer Graphics*, 21(4):163–169, July 1987. Proceedings of SIGGRAPH'87 (Anaheim, California, July 1987).

[NB93] Paul Ning and Jules Bloomenthal. An evaluation of implicit surface tilers. *IEEE Computer Graphics and Applications*, 13(6), November 1993.

[OSM93] Agata Opalash-Szwerbel and Steve Maddock. Implicit surfaces: Appearance, blending and consistency. In *Fourth Eurographics Workshop on Animation and Simulation*, Barcelona, Spain, 1993.

[PW89] Alex Pentland and John Williams. Good vibrations: Modal dynamics for graphics and animation. *Computer Graphics*, 23(3):215–222, July 1989. Proceedings of SIGGRAPH'89 (Boston, MA, July 1989).

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

[WG92] J. Wilhelms and A. Van Gelder. Octree for faster isosurface generation. *Transactions on Graphics*, 11(3):210–227, July 1992.

[WMG86] B. Wyvill, C. McPheeters, and G.Wyvill. Animating soft objects. *The Visual Computer*, pages 235–242, August 1986.

[WMW86] Geoff Wyvill, Craig McPheeters, and Brian Wyvill. Data structure for soft objects. *The Visual Computer*, pages 227–234, August 1986.

[WW89] Brian Wyvill and Geoff Wyvill. Field functions for implicit surfaces. *The Visual Computer*, 5:75–82, December 1989.

[Wyv93] Brian Wyvill. Metamorphosis of implicit surfaces. *Modeling, Visualizing, and Animating with Implicit Surfaces (SIGGRAPH'93 course notes Number 25, Anaheim, CA, USA)*, August 1993.

[ZJ91] Cornelia Zahlten and Hartmut Jurgens. Continuation methods for approximating isovalued complex surfaces. In *Eurographics'91*, pages 5–19, Vienne, Autriche, September 1991.