

12/07/2001

L'ANIMORPHOSE :  
MÉLANGE D'ANIMATION ET DE MÉTAMORPHOSE

Aurélien Barbier  
a-barb97@bat710.univ-lyon1.fr  
aurelien.barbier@free.fr

sous la direction d'Eric Galin, maître de conférence  
egalin@ligim.univ-lyon1.fr

L.I.G.I.M.  
(Laboratoire d'Informatique Graphique, Images et Modélisation)

Université Claude Bernard Lyon 1  
43 boulevard du 11 Novembre 1918 – 69622 Villeurbanne Cedex – France

## Remerciements

Je tiens tout particulièrement à remercier très vivement Eric Galin pour le soutien et les encouragements qu'il me prodigue depuis deux ans et sans qui je n'aurais sans doute pas fait ce DEA. Les longues conversations informelles que nous avons eues ont été pour moi une grande source d'enrichissement ; malgré une certaine persécution :-) ses conseils de rédaction pour ce mémoire ont été très instructifs.

Je voudrais également remercier Samir Akkouche d'avoir accepté d'être relecteur de ce mémoire.

Merci aussi à mes camarades de DEA au laboratoire pour avoir sacrifié un peu de leur temps de travail afin de m'escorter en pause lorsque mon bouillonnement cérébral frolait l'éruption.

Enfin, je souhaite remercier très chaleureusement Rami Hérelilier qui m'a permis de monopoliser un grand nombre de machines la nuit afin de réaliser les *animorphoses* que je présente ici comme je les désirais.

# Table des matières

<b>Introduction</b>	<b>5</b>
<b>1 Systèmes d'animation et de métamorphose</b>	<b>7</b>
1 Systèmes d'animation : un survol . . . . .	7
2 Systèmes de métamorphose : un état de l'art . . . . .	8
2.1 Les approches surfaciques . . . . .	8
2.2 Les approches volumiques . . . . .	11
2.2.1 Utilisation d'échantillons tridimensionnels . . . . .	11
2.2.2 Utilisation de surfaces implicites . . . . .	13
3 Quelles méthodes de métamorphose pour l'animorphose ? . . . . .	13
<b>2 Le BlobTree</b>	<b>15</b>
1 Introduction . . . . .	15
2 Les feuilles du BlobTree . . . . .	15
3 Les nœuds du BlobTree . . . . .	17
3.1 Opérateurs de combinaison . . . . .	17
3.2 Opérateurs de déformation . . . . .	17
4 Implémentation du BlobTree . . . . .	18
<b>3 Mixage d'animation et de métamorphose</b>	<b>19</b>
1 Notations . . . . .	19
2 Rappels sur la métamorphose du BlobTree . . . . .	20
2.1 Mise en correspondance . . . . .	20
2.2 Caractérisation des primitives à squelette . . . . .	20
2.3 Caractérisation des nœuds . . . . .	21
2.4 Métamorphose de Bézier . . . . .	21
3 Le BlobTree animé . . . . .	21
3.1 Paramètres de contrôle . . . . .	21
3.1.1 Animation des primitives du BlobTree . . . . .	22
3.1.2 Animation des nœuds du BlobTree . . . . .	22

3.2	Contrôle du mouvement . . . . .	23
4	Animorphose du BlobTree . . . . .	23
4.1	Algorithme . . . . .	23
4.1.1	Métamorphose de mouvement . . . . .	24
4.1.2	Séquences d'animorphose . . . . .	25
4.2	Animorphose de Bézier . . . . .	26
	<b>Conclusions et Perspectives</b>	<b>29</b>
	<b>Bibliographie et Netographie</b>	<b>31</b>
	<b>Abstract et Résumé</b>	<b>38</b>

# Introduction

Les surfaces implicites constituent un modèle géométrique particulièrement puissant pour la modélisation de formes lisses. Elles ont été utilisées indifféremment pour modéliser des liquides visqueux et des organismes vivants comme des coquillages, des plantes, des animaux ou des êtres humains.

Depuis 1998, au sein du laboratoire, Eric Galin et Samir Akkouche travaillent sur un modèle unificateur, le BlobTree, en collaboration avec le professeur Brian Wyvill de l'université de Calgary. Le BlobTree est un arbre de construction hiérarchique de surface implicite dont les feuilles sont des primitives à squelette et dont les nœuds sont choisis parmi des opérateurs de combinaison booléens, des opérateurs de mélange, ainsi que des opérateurs de déformation et des transformations affines.

Ce modèle s'est avéré être très intéressant pour la métamorphose. En effet, la métamorphose de surfaces implicites résout les problèmes de topologie qui limitent la plupart des méthodes fondées sur d'autres modèles de données, ce qui leur donne un avantage certain. En outre, le BlobTree fait preuve d'une grande aptitude à un contrôle fin et intuitif de la métamorphose, notamment grâce aux repères locaux attachés aux squelettes des primitives et à la généralité de la forme intermédiaire qu'il produit lors de sa métamorphose.

Avec la métamorphose et la déformation, la modélisation en vue de l'animation est un des axes forts du laboratoire. Or si de nombreuses études sur l'animation et la métamorphose sont présentes dans la littérature, le mélange d'animation et de métamorphose n'a jamais été abordé. Nous proposons ici de définir le cadre de l'*animorphose*, c'est-à-dire le mixage d'animation et de métamorphose. L'objectif est par exemple de transformer un ours qui marche en un oiseau qui vole.

L'*animorphose* permettra de générer des effets spéciaux pour le cinéma, la publicité, les clips et les jeux vidéo. En effet, dans ces marchés il peut être très intéressant de pouvoir définir la métamorphose d'objets animés en tenant compte des mouvements initial et final. Les difficultés principales sont de garantir la cohérence de la forme intermédiaire au cours de la transformation et d'offrir un contrôle le plus précis et intuitif possible à l'animateur. En outre, la définition d'un modèle intermédiaire générique interne à la classe des BlobTrees est très intéressante puisqu'elle permet de réappliquer l'*animorphose* à ce modèle.

Après avoir fait un état de l'art des systèmes d'animation et de métamorphose (chapitre 1) puis présenté le modèle du BlobTree (chapitre 2), nous en définissons une extension permettant son animation (chapitre 3). Le système d'animation que nous présentons est purement cinématique car dans l'objectif de l'*animorphose*, la transgression des lois de la dynamique au cours de la métamorphose interdit l'application d'un système physique. Afin de pouvoir définir des objets animés complexes, nous enrichissons également le Blobtree de nouvelles primitives, étendant ainsi les classes des primitives des polytopes aux squelettes volumiques convexes et aux splines. Nous rappelons ensuite la technique de métamorphose du BlobTree puis nous définissons la métamorphose de nos nouvelles primitives. Enfin, nous proposons une méthode originale de métamorphose d'objets animés qui conserve la propriété de généralité du modèle intermédiaire ce qui nous permet de définir l'*animorphose de Bier* dont les éléments de contrôle sont des *BlobTrees animés*.



# Chapitre 1

## Systèmes d'animation et de métamorphose

Actuellement, la métamorphose est principalement utilisée dans la production d'images de synthèse pour le cinéma, la publicité, les clips et les jeux vidéo. Les techniques existantes réalisent la métamorphose d'objets fixes. Dans ce cadre, une évolution vers la métamorphose d'objets animés serait très bénéfique puisqu'elle permettrait des effets spéciaux plus spectaculaires et mieux intégrés à la scène.

La métamorphose consiste à transformer une forme en une autre. Il existe évidemment une infinité de solutions à ce problème dont aucune n'est intrinsèquement plus juste qu'une autre. Cependant, un consensus se dégage pour tendre vers des métamorphoses les plus lisses possibles préservant au maximum la topologie et la géométrie des formes métamorphosées ; on parle alors de préserver la cohérence de forme. Ainsi, on considère généralement qu'une métamorphose est convaincante s'il n'existe pas d'étape intermédiaire informe. Pour autant, la qualité reste une notion éminemment subjective.

En vue de l'*animorphose*, nous nous intéressons ici aux différents systèmes d'animation et de métamorphose qui existent, ce qui nous permettra de choisir un modèle de données. Pour cela, nous porteront un intérêt particulier au contrôle de ces transformations.

### 1 Systèmes d'animation : un survol

Les systèmes d'animation existants [78] peuvent être classés en deux catégories principales : les systèmes physiques et les systèmes descriptifs. Dans les systèmes physiques, l'animation est uniquement régie par les lois de la physique qui tendent vers l'équilibre de la somme des forces s'appliquant à un objet avec le produit masse-accélération. Afin d'y parvenir, on peut recourir à la *dynamique directe* [21, 64, 31] pour déterminer le mouvement à partir des forces ou à la *dynamique inverse* [27, 15, 32, 30] pour calculer les forces qui génèrent un certain mouvement. Une variante est l'application à un système de particules des lois de la physique déterminant leur mouvement, les particules étant ensuite habillées d'une surface implicite pour la visualisation [21, 27, 39, 29]. L'inconvénient majeur de ces systèmes est qu'ils impliquent la connaissance et la gestion des forces qui s'appliquent aux objets ainsi que de bonnes connaissances biomécaniques lorsque l'on ne recourt pas aux systèmes de particules. En outre, ces méthodes sont extrêmement coûteuses puisqu'elles imposent à chaque étape la résolution de grands systèmes d'équations fortement paramétrées.

A l'opposé de l'approche physique, les systèmes descriptifs [26, 86] consistent en la caractérisation du mouvement à l'aide d'étapes clés déterminées par l'animateur, l'interpolation de ces étapes définissant le mouvement dans sa globalité. La *cinématique* est beaucoup plus simple à gérer que la dynamique d'autant qu'elle permet un contrôle précis de l'utilisateur, mais elle est soumise à la qualité de l'interpolation et peine à reproduire des animations physiquement réalistes. La qualité dépend donc fortement des compétences de l'animateur.

A mi-chemin entre systèmes physiques et descriptifs, des systèmes hybrides tentent de concilier la simplicité et le contrôle de la cinématique avec les résultats réalistes de la dynamique. Plusieurs approches

sont proposées dans cette optique et reposent sur l'utilisation de paramètres physiques afin de contraindre le mouvement caractérisé par les étapes clefs. L'utilisation de contraintes dynamiques [82, 70] n'est pas très bénéfique parce qu'elle impose à nouveau la gestion de nombreuses inconnues et peu ne pas conduire à une solution acceptable si les collisions et contacts entre objets ne sont pas modélisés par des contraintes de position physiquement valables.

Les systèmes cinématiques à validation dynamique [62, 63] offrent à cet égard des atouts intéressants puisque les lois de la dynamique sont uniquement utilisées afin de garantir la validité de l'animation déterminée selon la cinématique. Généralement, c'est donc la dynamique inverse qui est utilisée afin de calculer les forces mises en jeu puis de vérifier leur validité, permettant ainsi de modifier l'animation le cas échéant. Le coût de cette technique est bien moindre puisque la validation n'est pas nécessairement effectuée sur tout le modèle ; par exemple, on ne validera l'animation d'un marcheur montant un escalier qu'en garantissant l'absence d'interpénétration d'une de ses jambes avec les marches. À l'inverse, l'utilisation d'actionneurs vise à modéliser l'action des muscles en ajoutant un contrôleur dynamique à la cinématique puis à utiliser la dynamique directe afin de calculer l'animation finale. Cependant, là encore cette méthode impose de bonnes connaissances biomécaniques et s'avère coûteuse.

Enfin, l'édition de mouvements consiste à modifier une animation réelle capturée à l'aide de capteurs afin de l'adapter au mouvement désiré [69, 47, 85, 87, 96, 6]. Cette technique, très exploitée dans les jeux vidéo, souffre principalement de son inaptitude à générer des animations convaincantes lorsque le but et l'animation source sont trop dissemblables.

Tous les systèmes d'animation que nous avons vu sont indépendants du modèle de données. Pour autant, les surfaces implicites sont plus adaptées à la détection de collisions que les modèles surfaciques ou les échantillons de données 3D (voxels). Bien que la métamorphose puisse être considérée comme une animation par étape clef très particulière, nous la considérerons à part de par les changements de topologie et de géométrie extrêmes qui peuvent survenir.

## 2 Systèmes de métamorphose : un état de l'art

La métamorphose de formes tridimensionnelles est appréhendée selon deux approches totalement différentes : soit on effectue la métamorphose dans l'espace image (2D) à partir des projections des deux formes ou de dessins [50, 4], soit la métamorphose est effectuée dans l'espace des données (3D) puis on en tire une image. Dans l'optique de l'*animorphose*, la première approche apparaît clairement inadaptée. Nous allons donc nous intéresser aux différentes méthodes de métamorphose dans l'espace des données.

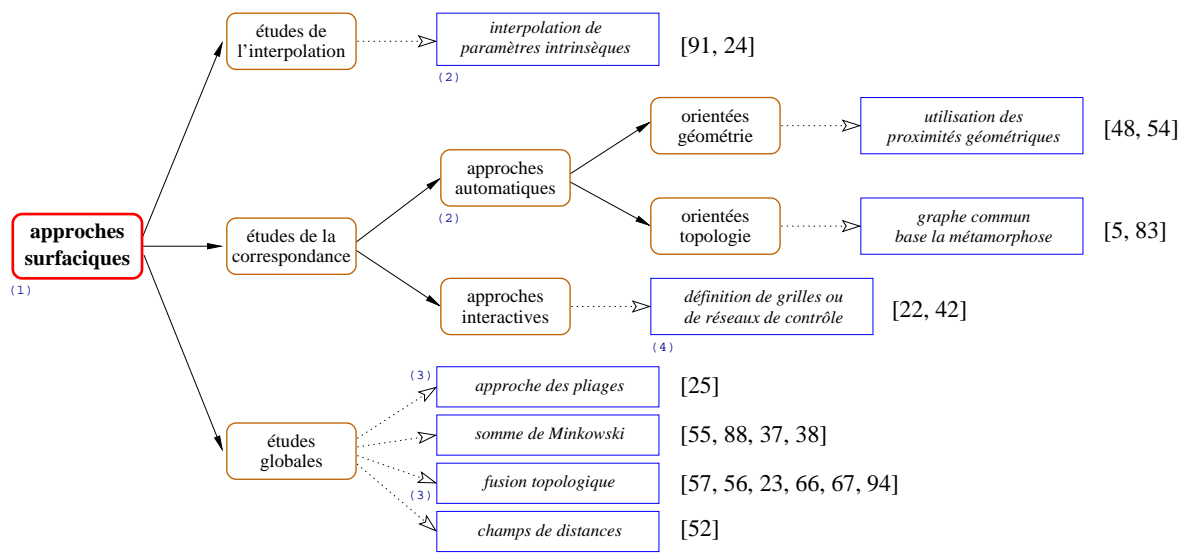
En nous appuyant sur la classification dressée par Lazarus et Verroust [65] ainsi que sur l'étude des méthodes de métamorphose volumique de Chen, Jones et Townsend [16], nous allons dresser une taxonomie des techniques de métamorphose en fonction du modèle des données manipulées puis des types d'algorithmes qui sont appliqués. Nous discuterons alors des avantages et des inconvénients de chaque méthode en fonction du type d'application que l'on désire réaliser, en l'occurrence des *animorphoses*.

### 2.1 Les approches surfaciques

La métamorphose d'objets maillés peut-être décomposée en deux phases : une étape de mise en correspondance des sommets des maillages et une étape d'interpolation définissant la trajectoire entre chaque paire de points. Le problème principal est celui de la mise en correspondance des maillages. Le plus souvent, l'interpolation est réduite à une interpolation linéaire. Une taxonomie des parutions sur la métamorphose de formes définies à l'aide de modèles surfaciques est schématisée dans la figure 1.1.

### L'interpolation de paramètres intrinsèques

Sun, Wang et Chin [91] se sont concentrés sur le problème de l'interpolation et considèrent que l'on possède deux maillages isomorphiques de polyèdres quelconques. Le but est de définir une transformation invariante aux mouvements solides à l'aide d'un paradigme de propagation pour interpoler les normales des faces. À partir de ces normales interpolées, les positions des sommets sont définies selon la même méthode de propagation. Il est notable que les normales des faces du polyèdre interpolé sont



- (1) il est relativement aisé de définir une structure de données hiérarchique (LOD) intéressante pour l'animation  
 (2) le contrôle est trop pauvre pour un passage à l'animation  
 (3) le champ d'application est trop restreint  
 (4) le contrôle, déjà lourd pour des objets fixes, semble interdire l'application à des objets animés

**Figure 1.1.** Une taxonomie des m´etamorphoses pour des objets surfaciques.

différentes des normales interpolées. La méthode peut être entièrement automatique. Dans le cas contraire, le contrôle est restreint à la phase d'initialisation de la propagation et consiste à choisir deux faces adjacentes et deux sommets de la première face sur chaque polyèdre. Du fait de l'utilisation de propagation, la méthode est fortement soumise aux approximations numériques.

Delingette, Watanabe et Suenaga [24] proposent une approche physique fondée sur la minimisation d'une fonction potentiel afin de déformer un maillage selon des contraintes prédéfinies. Les objets sont tout d'abord approximés par des maillages simples obtenus par déformation d'une sphère. L'utilisateur contrôle le processus de métamorphose en plaçant et étirant le maillage initial de chaque objet. Des opérateurs Eulériens sont utilisés pour adapter la connectivité du maillage au cours de la transformation qu'il est possible de contrôler en contraignant un point lorsque l'énergie atteint un minimum local. Puisque les objets sont re-maillés, tous les objets définis par un modèle surfacique peuvent être gérés. La géométrie des objets est ensuite caractérisée par un ensemble de paramètres intrinsèques adaptés à de tels maillages. La métamorphose se fait alors en ajoutant les contraintes qui contrôlent la transformation des paramètres intrinsèques.

### Mise en correspondance automatique

Les différentes méthodes de mise en correspondance automatique des maillages que nous allons présenter privilégient soit la géométrie, soit les relations topologiques.

Hong, Magnetat-Thalmann et Thalmann d'une part [48], puis Kanai, Suzuki et Kimura d'autre part [54], ont choisi l'approche géométrique. Hong et al. travaillent uniquement sur les facettes des maillages en effectuant la mise en correspondance par minimisation des distances de leurs centroïdes. Cette technique ne nécessite aucune relation d'adjacence entre les faces d'un même objet et est entièrement automatique. Au cours de la métamorphose, le nombre de facettes est le maximum de celui des deux maillages. Cependant, les résultats ne sont satisfaisants que pour des objets de formes similaires. De leur côté, Kanai et al. projettent chaque maillage dans un disque unitaire à l'aide de cartes harmoniques. La correspondance se fait alors par recouvrement des projections. Cette méthode s'applique à des polyèdres homéomorphiques à un disque ou à une sphère en les découpant en deux parties. Le seul contrôle offert à l'utilisateur est la rotation des disques autour de leur centre avant l'étape de mise en correspondance.

L'approche topologique est une autre option. Bethel et Uselton [5] définissent un super-maillage qui peut être réduit à l'un ou l'autre des deux maillages initiaux par fusion d'arêtes ou de faces. Ce super-

maillage est construit en parcourant simultanément les graphes d'aux des deux maillages et en ajoutant les sommets et faces nécessaires afin de maintenir la correspondance. Cette technique s'applique à des polyèdres orientés ayant le même type de topologie. Le contrôle est restreint au choix des deux faces correspondantes initiales. Cette méthode est purement topologique et ne tient pas compte de la géométrie des objets. En conséquence, le résultat peut être assez décevant pour des objets non topologiquement similaires. Parent [83], lui, définit une subdivision commune aux deux maillages en les découpant parallèlement en deux parties de manière récursive jusqu'à ce que chaque partie soit réduite à une face. Il obtient ainsi un même réseau de sommets, arêtes et faces pour chaque polyèdre. Cette méthode s'applique à des polyèdres homéomorphiques à une sphère. S'il le désire, l'utilisateur peut définir le découpage d'une partie à chaque étape mais généralement l'opération est entièrement automatique.

### Mise en correspondance interactive

De Carlo et Gallier [22] ont proposé une méthode de métamorphose de formes de topologies éventuellement différentes. Il appartient à l'utilisateur de définir une même grille de contrôle sur chacune des deux surfaces puis d'effectuer la mise en correspondance des cellules de ces deux grilles. La métamorphose se fait alors en deux étapes. Premièrement, une suite de formes et de grilles décrivant les changements de topologie est calculée selon la théorie de la chirurgie topologique en autorisant la création de faces dégénérées. Dans un second temps, la transformation est calculée par interpolation des étapes intermédiaires ainsi prédéfinies. Si le contrôle est précis et intuitif, il n'en est pas moins lourd même pour des objets relativement simples et nécessite de bonnes connaissances topologiques de l'utilisateur pour traiter les cas de changement de topologie.

Gregory et al. [42] ont une approche un peu similaire pour des polyèdres de même topologie représentés avec des arêtes ailées. Cette fois, l'utilisateur définit parallèlement sur chaque forme de base un réseau d'arcs en déterminant des points de contrôle. Les arcs sont créés de manière géodésique pour chaque paire de points de contrôle. L'utilisateur doit également définir les trajectoires d'interpolation pour chaque couple de sommets mis en correspondance entre les deux réseaux par des courbes de Bézier. Cette méthode qui fournit d'excellents résultats est le summum en terme de contrôle et de difficulté de l'exercer. A titre d'exemple, il a fallu plus de six heures aux auteurs pour créer une métamorphose entre un humain et un tricératops qui est cependant tout à fait convaincante.

### Polytopes en tant que polygones pliés

Demaine et al. [25] proposent une approche originale de métamorphose du cube. En effet, ils considèrent tout polyèdre convexe comme un polygone plié et effectuent la métamorphose de deux polytopes en métamorphosant leurs pliages. Cependant, il n'est pas encore démontré que la métamorphose de deux polytopes quelconques peut être réalisée par cette technique. En outre, il n'existe actuellement aucune caractérisation de la classe des polygones qui peuvent créer un polytope donné par des pliages. Pour autant, les auteurs fournissent des exemples intéressants.

### Somme de Minkowski

Kaul et Rossignac [55, 88] ont proposé une métamorphose à l'aide de sommes de Minkowski pour des polyèdres représentés par leur graphe d'adjacence. Rappelons que la somme de Minkowski de deux ensembles de points  $A$  et  $B$  est l'ensemble des points  $a + b$  où  $a \in A$  et  $b \in B$ . Cette méthode ne requiert aucune mise en correspondance et est totalement automatique. Les auteurs ont étendu cette technique à un ensemble de polyèdres en considérant une formulation de Bézier appliquée aux sommes de Minkowski. Galin et Akkouche [38, 37] utilisent également la somme de Minkowski pour effectuer la métamorphose de *blobs* à squelettes polyédriques convexes. Comme dans [100], ils définissent un graphe de correspondance puis, après l'avoir rendu bijectif par adjonction de composantes fantômes, ils expriment la forme intermédiaire comme un *blob* générique dont les composantes sont les interpolées de celles mises en correspondance dans le graphe bijectif. Les squelettes des composantes du *blob* intermédiaire générique sont obtenues par somme de Minkowski. En outre, ils étendent la méthode à une métamorphose de Bézier où les points de contrôle sont remplacés par des *blobs*. La qualité de la transformation dépend du graphe de correspondance défini par l'utilisateur mais un contrôle fin est rendu possible par l'utilisation de repères locaux [38].

## Fusion topologique

Différentes méthodes reposent sur une utilisation des projections des maillages sur des volumes de référence comme des sphères ou des cylindres. A la base de cette technique, on peut citer les travaux de Kent, Parent et Carlson [57, 56] qui consistent à projeter chaque maillage sur une sphère puis à les fusionner. La projection se fait radialement depuis un point central du maillage. Cette approche est donc limitée aux polyèdres étoilés et aux formes pouvant facilement être projetées sur une sphère. Le contrôle est restreint au choix du point central et à l'orientation de chaque forme. Decaudin et Gagalowicz [23] créent pour chaque objet une nouvelle forme dont le volume est égal à la somme des volumes en les étirant par rapport à un point central. Là aussi, la technique ne s'applique donc qu'à des polyèdres étoilés. La différence est que l'interpolation n'est pas linéaire pour chaque sommet mais que c'est l'interpolation du volume qui contraint l'interpolation des sommets. Le contrôle consiste uniquement en la détermination du point central des formes.

De leur côté, Lazarus et Verroust [66] ont proposé une approche similaire en projetant sur des cylindres des polyèdres étoilés autour d'une courbe. L'interpolation des points des maillages cylindriques approximant chaque objet est faite de façon radiale. Le principal problème est de déterminer la courbe directrice de chaque objet. Les auteurs proposent un algorithme de recherche automatique à partir d'un point de référence [67] mais ce faisant la méthode devient quasi automatique et le contrôle de la métamorphose est alors réduit à la modification des paramètres angulaires de la courbe et au déroulement de l'interpolation le long de l'axe. Pourtant, de l'aveu même des auteurs, il peut être extrêmement difficile de déterminer si un objet est étoilé autour d'une courbe et le cas échéant de la caractériser. La technique d'extraction automatique de cette courbe a été améliorée dans [94].

En conclusion, cette approche est intéressante mais limitée dans son champ d'application, c'est-à-dire aux formes étoilées. En outre, on peut lui reprocher le manque de contrôle qu'elle permet ou la difficulté de ce contrôle pour la méthode de Lazarus et Verroust.

## Champs de distances sur des triangulations

Kallmann et Oliveira [52] traitent conjointement les deux problèmes de mise en correspondance des sommets des maillages et de l'interpolation en utilisant des champs de directions définis sur ces maillages. Ils considèrent une version étirée de l'une des deux formes placée de manière adéquate à l'intérieur de l'autre puis déterminent un champ de vecteurs unitaires définis sur une tétraédrisation de l'espace entre les deux formes. Les champs ainsi définis ne doivent pas avoir de singularités et permettent alors d'effectuer une métamorphose préservant les topologies. Cependant, les modèles intermédiaires peuvent avoir un grand nombre de faces et il peut s'avérer difficile de déterminer les champs de direction. En outre, la complexité de la métamorphose de deux champs de direction n'a pas encore été établie et les algorithmes proposés sont très coûteux.

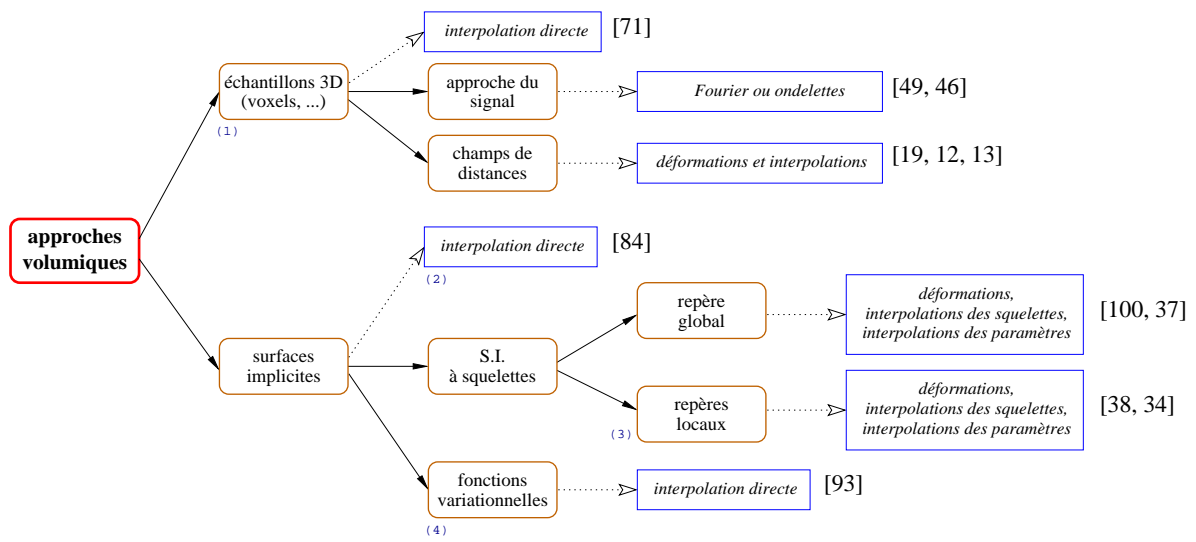
### 2.2 Les approches volumiques

La métamorphose d'objets volumiques peut être appréhendée selon deux types de modèles de données géométriques : les échantillons tridimensionnels (voxels, ...) et les surfaces implicites. La première solution s'affranchit de toute relation topologique et géométrique alors que la seconde offre un champ d'approches possibles beaucoup plus large. Afin de résumer toutes les méthodes que nous allons présenter, une taxonomie des parutions sur la métamorphose de formes définies à l'aide de modèles volumiques est schématisée dans la figure 1.2.

#### 2.2.1 Utilisation d'échantillons tridimensionnels

##### Interpolation directe d'échantillons 3D

Lerios, Garfinkle et Levoy [71] proposent de métamorphoser deux voxelisations adaptatives en les déformant afin qu'elles soient similaires, puis en les interpolant. Il s'agit en fait d'une extension tridimensionnelle de l'approche 2D de Beier et Neely [4]. L'utilisateur contrôle la transformation en mettant en correspondance des points, des segments, des rectangles ou des parallélogrammes rectangles. Si cette technique offre de bons résultats, elle est très gourmande en mémoire et en temps. D'autre part, de l'aveu



- (1) totalement inadapté à l'animation (coût de la mise à jour bien trop élevé)  
 (2) son manque de contrôle et son champ d'application restreignent l'application à l'animation  
 (3) l'utilisation de repères locaux fournit un grand plus dans le contrôle de la métamorphose en vue d'une animation  
 (4) contrôle déjà trop coûteux pour des objets fixes

**Figure 1.2.** Une taxonomie des métamorphoses pour des objets volumiques.

même des auteurs, l'extrême lourdeur de la mise en correspondance rend le contrôle long et réservé à des experts.

### Utilisation de transformées

Hughes [49] considère une transformée de Fourier des fonctions définissant les objets de base. He, Wang et Kaufman [46], eux, utilisent une transformée en ondelettes. Dans les deux cas, c'est une version discrète de la transformée qui est utilisée sur des échantillons de données 3D. L'interpolation est réalisée dans le domaine des fréquences spatiales. Il est notable que l'approche de He et al. établit la correspondance à un faible niveau de résolution. Ces deux méthodes sont entièrement automatiques et coûteuses en temps.

### Champs de distance

Cohen-Or, Levin et Solomovici [19] utilisent des objets définis par des ensembles hiérarchiques de fonctions de distance et de déformation. La métamorphose se fait alors en deux phases : une étape d'orientation et de légère déformation radiale des formes de base pour les rendre plus similaires, puis une étape d'interpolation des champs de distances ainsi modifiés. Cette méthode s'applique à des objets voxelisés et permet de contrôler la transformation par la mise en correspondance de points d'ancrages de certains voxels. Elle offre sans doute parmi les plus beaux résultats à ce jour mais au prix de longs temps de calcul.

Breen et al. [12] proposent une méthode de métamorphose de modèles solides exprimés avec des ensembles de niveaux. Afin de développer le champ d'application de leur technique, ils définissent également une méthode de métamorphose entre différents types de modèles géométriques (maillage, CSG, scan IRM) en les convertissant en leur modèle de données [13]. Leur métamorphose repose sur la déformation d'une forme en vue de la maximisation de sa similarité avec l'autre. Ce processus incrémental s'applique aux distances entre les voxels selon le niveau traité. Leur méthode impose un recouvrement partiel des deux formes à métamorphoser, ce qui peut être effectué automatiquement ou de manière interactive. Le contrôle est donc très restreint puisqu'il ne consiste qu'au positionnement relatif des deux objets. Pour autant, cette technique offre de bons résultats.

## 2.2.2 Utilisation de surfaces implicites

### Interpolation directe de surfaces implicites

Pasko et Savchenko [84] m'etamorphosent deux formes d'efinies à l'aide de surfaces implicites en interpolant lin'eairement ces fonctions. Cette m'ethode entièrement automatique fournit des r'esultats d'ecévants si les objets ne sont pas similaires et s'avère coûteuse.

### Surfaces implicites à squelettes

Les surfaces implicites à squelettes sont d'etermin'ees par le seuil d'un champ de potentiels autour de squelettes relativement à une fonction de distance. Wyvill et al [100] ont ouvert la voie de la m'etamorphose à l'aide de telles surfaces par la mise en correspondance de leurs squelettes puis l'interpolation des squelettes et des fonctions potentiel. Leur technique est restreinte à des squelettes ponctuels et repose essentiellement sur des heuristiques. Si le contrôle est ais'e pour des objets simples du fait du faible nombre d'el'ements du squelette, il en est tout autrement pour des objets complexes qui sont en outre difficiles à d'efinir.

Galin et Akkouche [37, 38] ont 'etendu la m'ethode à des squelettes poly'edriques convexes rendant ainsi la d'efinition d'objets complexes plus simple et par la même son côté intuitif au contrôle de la m'etamorphose. Cette g'en'eralisation s'est faite par l'utilisation de la somme de Minkowski pour interpoler les squelettes. L'introduction de repères locaux [38] aux 'el'ements du squelette puis celle du BlobTree [34] a accru le contrôle et la qualite des m'etamorphoses produites. En effet, la somme de Minkowski de deux primitives simples peut g'en'erer un polytope complexe, 'eventuellement de dimension sup'erieure dans le cas de primitives planes non parallèles. L'ajout de repères locaux permet donc une interpolation des squelettes optimale au sens de Minkowski ce qui 'elimine les objets interm'ediaires informes qui pouvaient apparaître. L'extension de la classe des primitives du squelette aux squelettes volumiques non poly'edriques [34] font de cette approche une des plus simplement contrôlables pour des objets de g'eom'etrie quelconque. En outre, la construction d'une forme interm'ediaire g'en'erique autorise la cr'eatation de *m'etamorphoses de Bézier*, ce qui permet un contrôle de haut niveau de la transformation.

### Utilisation de fonctions variationnelles

Turk et O'Brien d'efinissent leurs surfaces implicites variationnelles [92] à partir de points de la surface, des normales en ces points et de la localisation de l'int'erieur caract'eris'ee par un potentiel positif. S'il semble que la d'efinition de formes complexes peut être lourde, le contrôle, en revanche, est assez simple puisqu'il consiste en l'ajout de points de contrôle et la modification de normales. Pour d'efinir la m'etamorphose de deux objets de dimension  $N$  à l'aide de surfaces variationnelles [93], ils considèrent deux hyperplans parallèles comprenant les contraintes respectives de chacune des deux formes de base puis d'efinissent une surface variationnelle de dimension  $N + 1$  par une technique d'interpolation. La m'etamorphose se fait donc dans une dimension sup'erieure à celle des objets initial et final. La forme interm'ediaire de la m'etamorphose est alors la surface de dimension  $N$  d'efinie par la coupe de la surface dans l'espace de dimension  $N + 1$  selon un hyperplan parallèle aux deux hyperplans de base.

Cette technique combine les deux 'etapes de cr'eatation des surfaces implicites à m'etamorphoser et d'interpolation de celles-ci. Les auteurs indiquent que l'on peut contrôler la transformation à l'aide de formes d'influence mais ce contrôle n'est pas ais'e. En outre, la dimension dans laquelle la m'etamorphose sera ex'ecut'ee augmente lin'eairement avec le nombre de formes d'influence. Ainsi, cette m'ethode qui fournit des r'esultats très int'ereissants s'avère coûteuse pour des objets 'evolues et ardue à contrôler.

## 3 Quelles méthodes de métamorphose pour l'animorphose ?

A la lecture des diff'erentes m'ethodes propos'ees dans la litt'erature, il apparait qu'aucune ne s'applique à des modèles anim'es. A notre connaissance, il n'existe pas de technique de mixage d'animation et de m'etamorphose. Le but de ce travail est clairement de combler cette lacune.

Les approches surfaciques sont les plus utilis'ees à ce jour. La justification de cet 'etat de fait tient moins

à de quelconques avantages inhérents au modèle de données qu'aux possibilités offertes par les cartes graphiques et les modeleurs actuels pour afficher et manipuler ces données.

Les méthodes d'interpolation de paramètres intrinsèques de formes surfaciques ainsi que les techniques de mise en correspondance automatique n'offrent que peu de contrôle de la métamorphose et fournissent des résultats globalement décevants. La métamorphose par fusion topologique de maillages est restreinte à des polyèdres étoilés, au mieux autour d'une courbe, et difficile à contrôler comme les approches interactives que nous avons présentées. Enfin, l'utilisation de la somme de Minkowski n'est intéressante que pour des modèles surfaciques définis comme union de polyèdres convexes, ce qui est trop restrictif. Pour ces raisons, ces différentes méthodes sont inadaptées à la définition d'*animorphose* d'objets complexes.

De même, les méthodes opérant sur des échantillons 3D ne conviennent pas car les structures de données sont beaucoup trop lourdes à mettre à jour pour faire de l'animation.

A l'inverse des méthodes opérant sur des modèles surfaciques, les techniques utilisant des surfaces implicites ne sont pas limitées par des problèmes de topologie et permettent un contrôle de haut niveau. Une critique courante à l'égard des surfaces implicites est qu'elles sont difficiles à utiliser et inadaptées à la modélisation de formes complexes. Cependant, il semble que les mentalités sont en train d'évoluer et que les graphistes exploitent de plus en plus les potentialités des surfaces implicites proposées par les modeleurs actuels ([104]).

L'interpolation directe de surfaces implicites, de par son manque de contrôle et son faible champ d'application pour des résultats satisfaisants, est inadaptée à une extension à des objets animés complexes. D'autre part, les surfaces implicites à squelettes définies dans un repère global ne se prêtent pas à la définition des mouvements relatifs de leurs différentes composantes. Enfin, l'utilisation de fonctions variationnelles est trop coûteuse pour un contrôle fin de la métamorphose ce qui serait préjudiciable à une adaptation à l'*animorphose*.

Un paramètre extérieur à la métamorphose influe de manière importante sur le choix d'une approche plutôt qu'une autre : la visualisation. En effet, selon que l'on désire ou non effectuer une visualisation en temps interactif, certaines approches seront plus pertinentes que d'autres. Dans cette optique, un modèle surfacique semble plus adapté pour les possibilités qu'il offre, tant directement par sa propension à être facilement multirésolution, qu'indirectement par l'utilisation des cartes graphiques actuelles qui lui sont dédiées. De ce point de vue, les surfaces implicites sont en retrait car elles imposent une étape de triangulation en temps interactif, ce qui reste un problème ouvert bien qu'en évolution constante. Pour plus de détails sur les méthodes de maillage, on pourra se reporter à [72, 11, 77, 9, 35, 1]. Face à cet argument, les surfaces implicites à squelettes arguent d'une qualité supérieure pour une visualisation par lancer de rayons.

La facilité de définition d'objets complexes et surtout la généralité de la forme intermédiaire lors d'une métamorphose nous ont décidé à choisir le BlobTree, arbre de construction de surfaces implicites à squelettes avec des repères locaux, comme modèle de données pour l'*animorphose* d'objets complexes. Dans le chapitre 2, nous allons donc présenter le BlobTree, puis dans le chapitre 3 nous présentons en détail comment le métamorphoser et l'animer avant d'en définir l'*animorphose*.

## Chapitre 2

# Le BlobTree

Dans ce chapitre, nous présentons le modèle du BlobTree et précisons quelques notations qui seront utilisées dans le chapitre suivant sur la métamorphose et l'animation du BlobTree.

Le BlobTree a été proposé par Wyvill, Guy et Galin [98] afin de s'affranchir des limites du modèle des *blobs* à squelette dont le contrôle du mélange des différentes composantes ainsi que les possibilités de modélisation sont très restreintes.

### 1 Introduction

Une surface implicite  $\Sigma$  est définie comme l'ensemble des points  $\mathbf{x}$  de l'espace qui ont pour valeur un seuil  $T$  lorsqu'on leur applique une fonction  $f$  :

$$\Sigma = \{\mathbf{x} \in \mathbb{R}^3, f(\mathbf{x}) = T\} \quad (2.1)$$

Le *BlobTree* est un arbre de construction hiérarchique de surface implicite. La fonction potentiel  $f$  qu'il définit résulte de la combinaison d'un ensemble hiérarchique de fonctions. Plus précisément, les feuilles sont des primitives à squelette autour duquel on applique une fonction potentiel paramétrée par la distance à ce squelette et les nœuds sont des opérateurs de combinaison de ces potentiels (opérateur de mélange, opérateurs booléens et opérateur de mélange généralisé) ainsi que des opérateurs de déformation (torsion, écrasement et transformations affines). La figure 2.1 présente un BlobTree correspondant à la modélisation d'un chandelier : il est constitué du mélange de sphères écrasées et de la torsion de l'union de trois arêtes.

Il est notable qu'il n'y a pas unicité de la modélisation en BlobTree d'une forme quelconque donnée. En effet, la décomposition tant au niveau des feuilles que des nœuds peut être appréhendée selon plusieurs optiques. Cependant, on s'efforcera généralement de minimiser le nombre de primitives et d'équilibrer l'arbre au maximum.

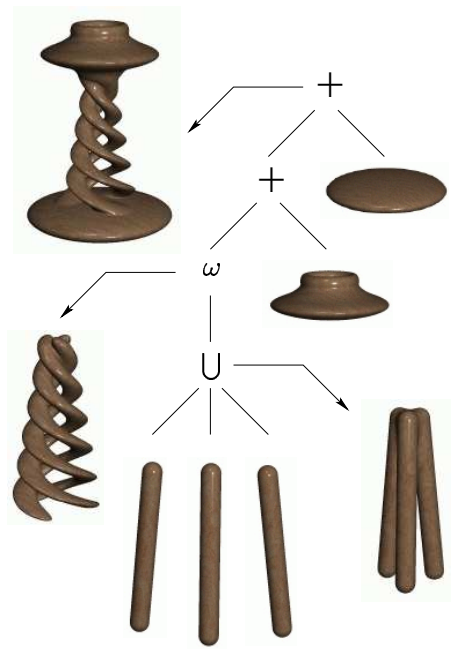
### 2 Les feuilles du BlobTree

On doit pouvoir calculer la distance au squelette d'une primitive, notée  $r$ , de tout point  $\mathbf{x}$  de l'espace afin de calculer l'intensité du potentiel et le gradient en ce point. Etant donné le squelette  $S_i$  d'une primitive  $i$ , une fonction de distance  $d_i$  à  $S_i$  et une fonction potentiel  $g_i$ , l'intensité au point  $\mathbf{x}$  est donnée par :

$$\forall \mathbf{x} \in \mathbb{R}^3 \quad f_i(\mathbf{x}) = g_i \circ d_i(\mathbf{x}) \quad (2.2)$$

Les fonctions de distance utilisées sont les distances associées à la norme  $\mathcal{L}_p$  notée  $\|\cdot\|_p$ , définie par :

$$\forall p \in \mathbb{R}_+^* \quad \forall (x, y, z) \in \mathbb{R}^3 \quad \|(x, y, z)\|_p = (|x|^p + |y|^p + |z|^p)^{1/p} \quad (2.3)$$



**Figure 2.1.** Arbre de construction d'un chandelier [34].

Dans la quasi totalité des cas, on se contentera d'utiliser la distance Euclidienne, soit  $p = 2$ . Pour  $p \neq 2$ , la distance correspondante n'est pas isotrope, ce qui implique que tous les calculs doivent être réalisés dans un repère propre à chaque squelette.

Afin d'accélérer les calculs d'intensité de potentiel et de gradient, les fonctions potentiel  $g$  que nous utiliserons sont des fonctions en  $r^2$ . Différentes fonctions potentiel ont été proposées dans la littérature ; nous allons les présenter sous forme normalisée, c'est-à-dire avec  $R = 1$  et  $I = 1$  (donc  $r \in [0; 1]$ ). Wyvill, Pheeters et Wyvill [102] ont proposé d'utiliser une fonction cubique en  $r^2$  :

$$g(r) = -\frac{4}{9}r^6 + \frac{17}{9}r^4 - \frac{22}{9}r^2 + 1 \quad (2.4)$$

Une autre famille de fonctions potentiel permettant une continuité de classe  $\mathcal{C}^{n-1}$  a été proposée par Murakami et Ichihara [79] et s'écrit :

$$\forall n \geq 2 \quad g(r) = (1 - r^2)^n \quad (2.5)$$

Enfin, Blanc et Schlick [8] ont proposé des fonctions sous forme de fractions rationnelles en  $r^2$  paramétrées par un coefficient de raideur  $k$  :

$$\begin{cases} g(r) = 1 - \frac{9r^4}{k + \left(\frac{9}{2} - 4k\right)r^2} & 0 \leq r^2 \leq \frac{1}{4} \\ g(r) = \frac{(1 - r^2)^2}{\frac{3}{4} - k + \left(\frac{3}{2} + 4k\right)r^2} & \frac{1}{4} \leq r^2 \leq 1 \end{cases} \quad (2.6)$$

Conformément à la définition (2.2) nous aurons besoin de connaître, en plus de la fonction potentiel, le rayon d'influence  $R$  et l'intensité maximale  $I$  qui s'appliquent au squelette.

Les primitives à squelette ponctuel sont très fréquemment utilisées parce qu'elles permettent de calculer aisément la distance d'un point au squelette. Cependant, l'inconvénient majeur est que la modélisation d'objets complexes nécessite un très grand nombre de telles primitives. Aussi, Galin et Akkouche [37]

ont étendu la classe des primitives aux polyèdres convexes. Ce faisant, il est désormais possible de garantir le potentiel à l'intérieur d'un volume et donc de mieux contrôler la modélisation d'objets complexes tout en réduisant de manière significative le nombre des primitives nécessaires.

Afin d'étendre les possibilités de modélisation, nous avons implémenté plusieurs primitives à squelettes volumiques convexes non polyédriques comme le cône et le tore, ainsi que les splines qui permettent d'étendre la classe des primitives à des squelettes non convexes. L'extension de la classe des primitives a toujours pour but d'accroître les possibilités de modélisation de formes et leur qualité ou de simplifier leur réalisation pour éviter par exemple d'avoir à recourir à des opérateurs de déformation coûteux. A titre d'exemple, si on considère la queue d'une souris, une seule spline suffit à la modéliser alors qu'il fallait jusqu'à présent un grand nombre de segments ou une quantité excessive de points pour arriver à des résultats moins bons visuellement.

Comme nous le verrons dans la section 2 du chapitre 3, l'utilisation de squelettes plus complexes nous conduira à étudier la transformation non triviale de ces squelettes entre eux pour la métamorphose.

### 3 Les nœuds du BlobTree

Les nœuds du BlobTree [98] sont les opérateurs de combinaison de potentiels et les opérateurs de déformation dont font partie les transformations affines. Les opérateurs de combinaison de potentiels contrôlent les contributions des différentes primitives. De nombreuses définitions de tels opérateurs sont possible. Dans notre implémentation, nous utilisons ceux qui ont été présentés par Galin et Akkouche [34].

#### 3.1 Opérateurs de combinaison

L'*opérateur de mélange* permet de définir une jonction lisse entre les contributions des différents squelettes. Il s'agit du seul opérateur de combinaison qui était utilisé pour les *blobs* à squelettes. Il est défini simplement par :

$$f_{A+B}(\mathbf{x}) = f_A(\mathbf{x}) + f_B(\mathbf{x}) \quad (2.7)$$

Les *opérateurs booléens* permettent d'appliquer la CSG<sup>1</sup> au BlobTree et donc de définir entre autre des jonctions discontinues entre les composantes. Ils peuvent être définis par l'utilisation de **R**-Fonctions [84] ou plus simplement par<sup>2</sup> :

$$\begin{aligned} f_{A \cup B}(\mathbf{x}) &= \max(f_A(\mathbf{x}), f_B(\mathbf{x})) \\ f_{A \cap B}(\mathbf{x}) &= \min(f_A(\mathbf{x}), f_B(\mathbf{x})) \\ f_{A \setminus B}(\mathbf{x}) &= \min(f_A(\mathbf{x}), 2T - f_B(\mathbf{x})) \end{aligned} \quad (2.8)$$

Enfin, l'*opérateur de mélange généralisé* se situe à mi-chemin entre les deux types d'opérateurs précédents et permet de choisir le degré de lissage des jonctions qui se font entre les composantes. Il sera notamment très utile pour métamorphoser un opérateur de mélange en opérateur d'union ou d'intersection. Il est défini par :

$$\forall p \in \mathbb{R}_+^* \quad f_{A*B}(\mathbf{x}) = (f_A(\mathbf{x})^p + f_B(\mathbf{x})^p)^{1/p} \quad (2.9)$$

#### 3.2 Opérateurs de déformation

Comme leur nom l'indique, les *opérateurs de déformation* permettent de déformer une partie d'une surface implicite en déformant l'espace autour de primitives. Ils ont été définis par Barr [2]. Pratiquement, on a :

$$f_\omega(\mathbf{x}) = f \circ \omega^{-1}(\mathbf{x}) \quad (2.10)$$

<sup>1</sup>Constructive Solid Geometry.

<sup>2</sup>la différence est un peu moins évidente et dépend du seuil du BlobTree. On a  $A \setminus B = A \cap \bar{B}$  et  $f_{B-} = 2T - f_B$

<sup>3</sup>pour  $p = 1$  on retrouve l'opérateur de mélange, pour  $p \rightarrow +\infty$  il est équivalent à l'opérateur d'union et lorsque  $p \rightarrow -\infty$  il est équivalent à l'opérateur d'intersection.

Dans notre implémentation, la torsion et l'écrasement ont été implémentés. De nombreux autres opérateurs peuvent être définis [90]. Bien que les *transformations affines* puissent être considérées comme des déformations particulières, elles sont implémentées à part afin de permettre une optimisation par concaténation. Les transformations affines implémentées sont une combinaison de translations, de rotations et d'homothéties selon l'ordre suivant [36] :

$$\mathbf{A} = \mathbf{T} \circ \mathbf{R} \circ \mathbf{H} \quad (2.11)$$

Plus précisément, nous utilisons une matrice homogène pour stocker les transformations affines et permettre ainsi la concaténation simplement par produit des matrices des différentes composantes de la transformation.

#### 4 Implémentation du BlobTree

Nous ne présentons ici que le schéma organisationnel de notre implémentation en C++. Rappelons que son nœud racine suffit à déterminer un BlobTree.

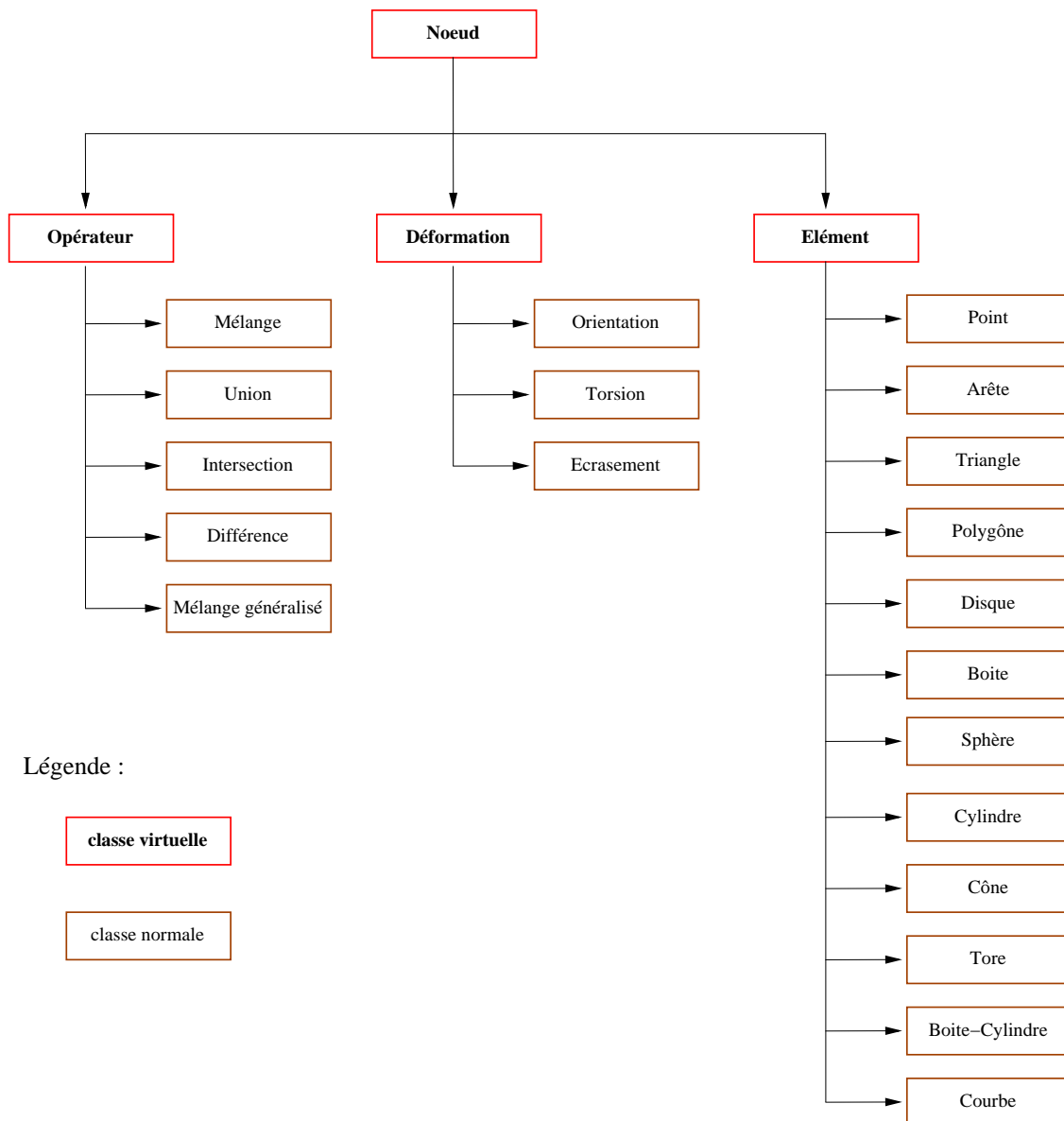


Figure 2.2. Graphe d'héritage simplifié des classes du BlobTree.

## Chapitre 3

# Mixage d'animation et de métamorphose

Dans ce chapitre, nous présentons notre méthode originale permettant de mixer métamorphose et animation afin de définir des *animorphoses* d'objets complexes. Notre contribution est double : d'une part nous proposons un modèle d'animation cinématique pour le BlobTree, et d'autre part nous montrons comment intégrer cette méthode d'animation dans la métamorphose. Ce chapitre s'organise en trois parties : nous rappelons tout d'abord la méthode de métamorphose du BlobTree et présentons ensuite une extension du modèle du BlobTree à des primitives à squelette volumique complexe éventuellement non convexe ainsi qu'un modèle de métamorphose pour ces squelettes, puis, nous proposons une extension du modèle du BlobTree pour l'animation, et enfin nous présentons notre méthode d'*animorphose*.

Puisque nous cherchons à mélanger animation et métamorphose, il faut définir ces deux notions dont la frontière est assez floue. Nous parlerons d'*animation* pour définir l'ensemble des mouvements solides (translations, rotations) qui peuvent s'appliquer à un objet ou à une partie de celui-ci. Le terme de *déformation* sera consacré à l'ensemble des opérations modifiant la géométrie d'un objet mais qui conservent sa topologie (homothéties, étirements, torsions). Enfin, la *étamorphose* désignera toute transformation générale d'un objet initial en un objet final engendrant des changements de géométrie et/ou de topologie. Des définitions précises de ces termes seront données dans la section sur l'*animorphose* du BlobTree.

### 1 Notations

Dans cette partie nous précisons quelques notations utiles. Soit  $A$  un BlobTree fixe et  $\mathcal{A}$  ses composantes, que ce soit des nœuds ou des feuilles. Soit  $(A, \mathcal{M}_A)$  un BlobTree  $A$  animé selon un mouvement noté symboliquement  $\mathcal{M}_A$ . Dans un souci de simplification, on notera plus simplement  $A$  ce même *BlobTree animé* lorsqu'il n'y aura pas de confusion possible avec un BlobTree fixe. Nous noterons respectivement  $\mathcal{P}_{A_i}$  et  $\mathcal{P}_{\mathcal{M}_{A_i}}$  les vecteurs des paramètres définissant le mouvement de la composante  $A_i$  d'un Blobtree animé  $(A, \mathcal{M}_A)$ .

Enfin, soit  $\Delta$  l'opérateur de métamorphose et d'*animorphose*, le type de la transformation étant défini implicitement par les opérandes auxquelles il s'applique. Le graphe de correspondance entre les composantes des BlobTrees initial  $A$  et final  $B$  servant de base à la construction du BlobTree générique  $C$  sera noté  $\mathcal{G}(A \rightarrow B)$ .  $C(t)$  désignera une instance de  $C$  à l'instant  $t$ .

Pour la métamorphose, nous considérerons un BlobTree initial  $A$  de  $n_A$  composantes, un BlobTree final  $B$  de  $n_B$  composantes et le BlobTree générique interpolé  $C$  de  $n_C \leq n_A \times n_B$  composantes. Pour deux composantes  $A_i$  et  $B_j$  apparues dans le graphe de correspondance bijectif  $\mathcal{G}(A \rightarrow B)$  de  $A$  et de  $B$ , on notera alors :  $C_{ij}(t) = \Delta(A_i, B_j)$ .

Par analogie, pour l'*animorphose*, nous considérerons le BlobTree générique interpolé  $(C, \mathcal{M}_C)$  de  $n_C \leq n_A \times n_B$  composantes :  $(C, \mathcal{M}_C) = \Delta((A, \mathcal{M}_A), (B, \mathcal{M}_B))$  dont  $(C(t), \mathcal{M}_C(t))$  est une instance.

## 2 Rappels sur la métamorphose du BlobTree

La métamorphose de *blobs* à squelettes a été introduite par Galin [36] puis améliorée par Galin et Akkouche [37, 38], notamment afin d'empêcher l'apparition d'étapes intermédiaires informes et d'accroître le contrôle. La qualité d'une métamorphose étant une notion hautement subjective, le contrôle de la transformation est primordial. Après la définition du modèle du BlobTree [98], Galin, Leclercq et Akkouche ont donc étendu leur méthode de métamorphose au BlobTree [34]. Nous allons présenter brièvement cette méthode dans cette partie.

**Définition :** Un *BlobTree métamorphosé* est un modèle générique de BlobTree issu d'un graphe de correspondance des composantes des BlobTrees initial et final.

### 2.1 Mise en correspondance

La première phase de la méthode est la mise en correspondance des nœuds des BlobTrees initial et final. L'utilisateur associe à chaque primitive du BlobTree initial un ensemble de primitives du BlobTree final. Des composantes de chacun des BlobTrees peuvent également être mises en correspondance avec des primitives fantômes. Wyvill [99] a défini des algorithmes automatiques de mise en correspondance fondés sur des heuristiques mais les résultats ne sont pas toujours satisfaisants. Galin et Akkouche [38] ont proposé une approche semi-automatique qui consiste en la mise en correspondance d'ensembles de composantes des deux BlobTrees.

La mise en correspondance ne donnant généralement pas de graphe bijectif, il faut décomposer les nœuds et les feuilles des arbres de construction  $A$  et  $B$  afin d'obtenir un nouveau graphe bijectif. Grâce au nouveau graphe, on peut définir simplement chaque nœud et feuille  $C_j$  de l'arbre à l'aide de ses parents  $A_i$  et  $B_j$ .

### 2.2 Caractérisation des primitives à squelette

On cherche ici à construire la primitive  $C_{ij}$  interpolant deux primitives  $A_i$  et  $B_j$ . Rappelons brièvement qu'une primitive est caractérisée par un squelette  $S$ , une fonction potentiel  $g$ , une fonction distance  $d$  ainsi qu'une intensité maximale du potentiel  $I$  constante à l'intérieur du squelette et un rayon d'influence  $R$ . Métamorphoser des primitives revient à interpoler chacune de ces caractéristiques. Dans la pratique, on crée une primitive générique dont les caractéristiques sont les interpolées de celles de  $A$  et  $B$ .

Comme écrit dans [37], l'interpolation de deux squelettes polytopes se fait à l'aide de la somme de Minkowski, c'est-à-dire que le squelette de  $C_{ij}$  est défini par :

$$S_{C_{ij}(t)} = \{\alpha(t) S_{A_i} \oplus \beta(t) S_{B_j}, \alpha(t) + \beta(t) = 1\} \quad (3.1)$$

les fonctions  $\alpha(t)$  et  $\beta(t)$  caractérisant l'interpolation de  $A$  vers  $B$ . Dans notre implémentation,  $\alpha(t)$  et  $\beta(t)$  sont des splines, généralement cubiques. Les repères locaux permettent également de contrôler la transformation.

Nous avons étendu la métamorphose aux squelettes non polyédriques et à certains squelettes non convexes. Rappelons que nous avons étendu le modèle du BlobTree en ajoutant de nouvelles primitives à squelette volumique (sphère, cône, cylindre, tore) et des primitives à squelette particulier (disque, spline). Pour la métamorphose de ces squelettes, la caractérisation du squelette générique intermédiaire n'est pas aisée. Certes, pour la métamorphose de squelettes volumiques convexes non polyédriques, la somme de Minkowski fonctionne encore mais elle ne génère pas une forme géométrique triviale dont on peut calculer algorithmiquement la distance à un point de l'espace. Ceci nous a conduit à prévoir des squelettes de forme complexe comme la boîte-cylindre. Pour les squelettes plus complexes tels que la spline ou le disque, il faut prévoir des étapes de passage intermédiaires. Par exemple, on assurera la métamorphose d'un disque vers un segment en passant par un cylindre relativement aisément mais la transformation d'un disque en spline est beaucoup plus délicate à définir.

L'interpolation directe de fonctions potentiel est une méthode générale mais elle donne des résultats peu satisfaisants car elle ne garantit pas la généralité de la fonction potentiel de la primitive  $f(t)$  et induit un surcroît de calcul. On préférera donc interpoler les paramètres de ces fonctions, à savoir le rayon d'influence  $R$ , l'intensité maximale  $I$  et éventuellement le coefficient de raideur  $k$  [37].

De même, la recherche d'une définition générique pour les fonctions distance nous fait préférer l'interpolation de leurs paramètres plutôt qu'une interpolation directe. Les fonctions distances étant toutes des distances  $\mathcal{L}^p$ , nous interpolons le degré  $p$  selon une spline en fonction du temps.

## 2.3 Caractérisation des nœuds

La métamorphose de nœuds de même type se fait par interpolation selon des splines de leurs paramètres de définition. En revanche, la métamorphose de nœuds de types différents ne peut pas se faire directement. On se ramènera alors à une métamorphose de deux nœuds de même type en modifiant un des deux BlobTrees par introduction d'un nœud correspondant à l'élément neutre pour le type prioritaire.

## 2.4 Métamorphose de Bézier

La métamorphose de BlobTrees fournissant un modèle générique, il est possible de définir une *métamorphose de Bézier*. La métamorphose se fait alors dans l'espace des BlobTrees, les points de contrôle des courbes de Bézier étant remplacés par des BlobTrees de contrôle. Là encore on cherche à définir un modèle générique qui sera utilisé pour toute la transformation. Cependant, la difficulté est accrue par le fait qu'hormis les modèles initial et final, tous les BlobTrees de contrôle sont impliqués dans deux graphes de correspondance. Ceci influe en fait directement sur la décomposition en sous-graphes de correspondance en multipliant les contraintes, mais la technique reste globalement similaire. Les graphes de correspondance bijectifs étant établis, on se retrouve dans le cas classique de métamorphose de deux BlobTrees dont les arbres se superposent. On pourra en particulier se reporter à [36] pour plus de détails.

## 3 Le BlobTree animé

Dans cette partie, nous présentons le modèle du *BlobTree animé* qui constitue une extension du modèle du BlobTree original [98]. En effet, le modèle original du BlobTree ne définit que des objets fixes. Nous allons donc introduire le concept de *BlobTree animé* en tant qu'arbre hiérarchique de nœuds et de feuilles dont les paramètres sont fonction du temps. On appellera désormais *animation* toute transformation qui ne modifie ni le type des squelettes des primitives ni le type des nœuds de l'arbre. Dans la suite nous parlerons donc indifféremment d'animation à propos d'animation pure ou de déformation. À l'inverse, on appellera *métamorphose* toute transformation modifiant le type du squelette d'au moins une primitive ou le type d'au moins un nœud de l'arbre.

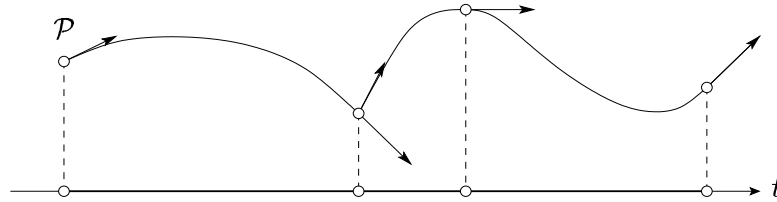
**Définition :** un *BlobTree animé* est une arborescence générique dont chaque nœud et feuille possède une liste de paramètres fonctions du temps mais dont les types des nœuds et des squelettes restent constants au cours du temps.

Le modèle d'animation que nous proposons est purement cinématique mais son implémentation reste ouverte à une extension vers un modèle à validation dynamique. Nous avons exclu l'idée d'un système d'animation physique tant il est évident qu'au cours de la métamorphose que nous souhaitons effectuer les lois de la physique n'ont plus de fondement. Chaque composante d'un *BlobTree animé* peut posséder une animation propre définie à l'aide d'étapes clefs donnant à un instant  $t_i$  des valeurs des paramètres qui caractérisent cette composante. Il faut noter que deux étapes clefs peuvent être définies à un même instant  $t_i$ , par exemple pour modéliser un rebond (positions identiques mais vitesses différentes). En outre, les instants clefs d'une composante ne sont pas nécessairement clefs pour une autre.

### 3.1 Paramètres de contrôle

Nous définissons un *BlobTree animé* en paramétrant chacune de ses caractéristiques en fonction du temps. Chaque composante d'un *BlobTree animé* possède un mouvement propre, ce qui permet de réduire la définition détaillée du *BlobTree animé* à celle de chacune de ses composantes qui sont soit des primitives, soit des opérateurs. Les paramètres à contrôler sont essentiellement des grandeurs scalaires (rayon d'influence, intensité maximale et coefficient de raideur d'une fonction potentiel, ...) ou des grandeurs vectorielles dans  $\mathbb{R}^3$  (position, vitesse et accélération).

L'interpolation entre deux étapes clés se fera par interpolation de ces paramètres. Pour cela, nous avons choisi d'utiliser des splines par morceaux pour les possibilités de contrôle intuitif et précis qu'elles offrent (Figure 3.1). Il revient à l'utilisateur de choisir pour chaque morceau un degré quelconque d'interpolation en fonction du contrôle qu'il désire effectuer et de la précision de sa définition du BlobTree. Par exemple, connaissant les positions et les vitesses d'un point aux instants 0 et  $1^1$ , la courbe de plus bas degré respectant ces contraintes est une cubique dont les coefficients sont déterminés par l'interpolation d'Hermite-Ferguson. Nous n'avons implémenté que les interpolations de degré 1, 3 et 5 correspondant respectivement à une définition d'un paramètre à partir des valeurs, des couples valeur-vitesse ou des triplets valeur-vitesse-accélération. Dans toute la suite nous considérerons le cas des Hermite-splines (degré 3) qui présentent le meilleur rapport qualité-coût.



**Figure 3.1.** Paramètres intrinsèques de l'animation définis en fonction du temps.

Finalement, le choix de cette méthode de définition du *BlobTree animé* nous permet de conserver le graphe d'héritage des classes de notre implémentation du BlobTree classique puisque la seule modification est la paramétrisation des caractéristiques de définition des composantes de l'arbre par le temps. Le graphe d'héritage simplifié du BlobTree présenté dans la figure 2.2 reste donc valable pour le *BlobTree animé*. Ainsi, une instance de *BlobTree animé* peut être considérée comme un blobTree classique. Nous allons maintenant expliciter les paramètres de définition des composantes du *BlobTree animé*.

### 3.1.1 Animation des primitives du BlobTree

Rappelons qu'une primitive de BlobTree est caractérisée par son squelette  $S$  et une fonction potentiel  $g$ . Une primitive de *BlobTree animé* possède donc un ensemble d'étapes clés aux instants  $t$  composées des paramètres de définition du squelette et de la fonction potentiel.

Les paramètres qui définissent un squelette sont par exemple le centre et le rayon pour une sphère ou deux sommets diagonaux pour une boîte. Ceux qui caractérisent une fonction potentiel sont le rayon d'influence, l'intensité maximale et éventuellement un coefficient de raideur lorsque l'on utilise les fonctions rationnelles de Blanc [8]. Une primitive de *BlobTree animé* possède également diverses méthodes qui, sur un intervalle temporel  $]t_1; t_2[$ , permettent de connaître les valeurs de ces grandeurs dépendantes du temps à un instant  $t$  donné.

### 3.1.2 Animation des nœuds du BlobTree

Les opérateurs booléens (union, intersection et différence) et l'opérateur de mélange sont les mêmes opérateurs que ceux du BlobTree classique. L'opérateur de mélange généralisé fait exception puisque son degré est paramétré par le temps.

Les opérateurs de déformation ont des paramètres de définition propres qui sont fonction du temps et possèdent donc un ensemble d'étapes clés caractérisant ces paramètres aux instants  $t$ . La torsion et l'écrasement qui sont implémentés ont été présentés par Barr [2]. La torsion est définie par l'axe de torsion représenté par ses deux extrémités  $a(t)$  et  $b(t)$  ainsi que par la vitesse angulaire de torsion  $\omega(t)$ . L'écrasement, lui, est caractérisé par les deux extrémités  $a(t)$  et  $b(t)$  de l'axe d'écrasement ainsi que par leurs rayons d'écrasement respectifs  $r_a(t)$  et  $r_b(t)$ .

Les transformations affines sont un peu plus délicates. Dans notre implémentation, une transformation affine est définie comme la combinaison d'homothéties, de rotations et de translations qui sont

<sup>1</sup>pour tout espace temporel de travail  $]t_1; t_2[ \neq \emptyset$  on peut se ramener au cas  $]0; 1[$  par une transformation affine

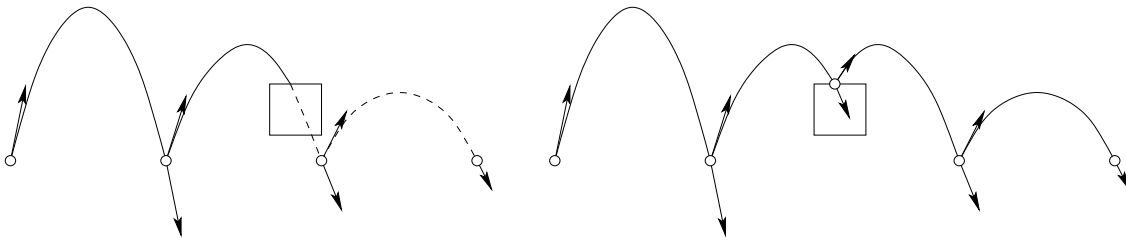
paramétrées par le temps :

$$\mathbf{A}(t) = \mathbf{T}(t) \circ \mathbf{R}(t) \circ \mathbf{H}(t) \quad (3.2)$$

Nous interpolons indépendamment chacune de trois composantes d'une transformation affine. Pour les translations et les homothéties nous procédons toujours par interpolation des paramètres selon des splines, mais pour les rotations cela n'est pas satisfaisant. En effet, l'utilisation des angles d'Euler pour gérer les orientations dans le *BlobTree* n'est pas transposable au *BlobTree animé* car ils supportent mal l'interpolation provoquant un effet dit de *gimbal lock* qui entraîne la perte d'un degré de liberté. Nous avons donc choisi d'utiliser les quaternions qui permettent une interpolation lisse à l'aide d'une fonction *slerp* présentée dans [89, 58, 76].

### 3.2 Contrôle du mouvement

Comme pour la métamorphose, le contrôle de l'animation est primordial. Avec le *BlobTree animé*, ce contrôle s'applique tout d'abord par la définition des paramètres intrinsèques de ses composantes ainsi que par le choix de la méthode qui sera utilisée pour les interpoler. Cependant, il faut souligner que ce contrôle est restreint à l'animation d'un objet indépendamment de l'environnement dans lequel il évolue. Aussi, nous allons examiner le problème de la correction de trajectoire.



**Figure 3.2.** La figure de gauche illustre une animation où un ballon traverse un obstacle au cours de sa trajectoire. A droite, l'adjonction d'une nouvelle étape clé suffit à corriger la trajectoire.

La définition par étapes clés d'une animation se faisant hors de tout environnement, des interopérations d'objets non désirées par l'utilisateur peuvent survenir, rendant alors l'animation globale non consistante. Pour ce type de problème, les splines qui définissent l'animation permettent un contrôle précis et intuitif. En effet, l'insertion de nœuds, la suppression et le déplacement de points de contrôle offrent un contrôle très fin. Dans notre exemple (Figure 3.2), la trajectoire du ballon est corrigée en ajoutant une nouvelle étape clé correspondant au rebond sur l'obstacle. Le reste de la définition reste inchangée et l'animation devient consistante.

## 4 Animorphose du BlobTree

Dans cette partie, nous cherchons à métamorphoser deux modèles animés  $(A, \mathcal{M})$  et  $(B, \mathcal{M}_B)$  de la manière la plus esthétique possible avec un contrôle maximal.

**Définition :** on appelle *BlobTree animorphosé* (ou *BlobTree animorphé*) de deux *BlobTrees animés*  $(A, \mathcal{M}_A)$  et  $(B, \mathcal{M}_B)$  un modèle générique de *BlobTree animé*  $(C(t), \mathcal{M}_C)$  où  $C(t)$  est le modèle générique transformant  $A$  en  $B$  et où  $\mathcal{M}$  est une description de mouvement transformant  $\mathcal{M}_A$  en  $\mathcal{M}_B$ .

### 4.1 Algorithme

Puisque du point de vue de l'implémentation, la définition du *BlobTree animé* modifie pas la structure du modèle, l'*animorphose* est quasiment identique à la métamorphose puisqu'elle est définie comme une métamorphose de *BlobTrees* classiques accompagnée d'une métamorphose de leurs mouvements respectifs. Dans la pratique, il faut donc construire un *BlobTree animé* générique dont les paramètres de définition sont déterminés par interpolation de  $\mathcal{M}_A$  et  $\mathcal{M}_B$ . Schématiquement, la méthode que nous proposons correspond à l'algorithme suivant :

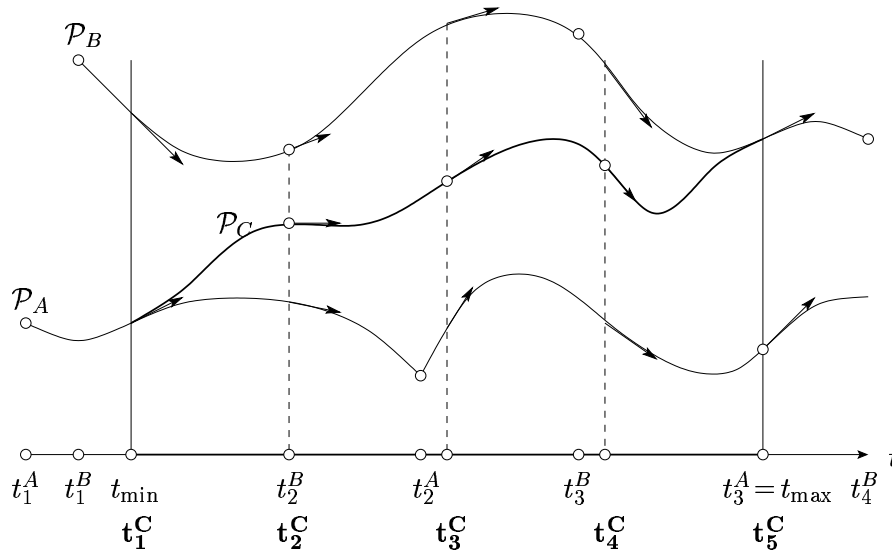
1. Définition du graphe de correspondance des composantes :  $\mathcal{G}(A \rightarrow B)$ .
2. Construction du BlobTree métamorphosé générique  $C(t)$  à partir du graphe  $(\mathcal{A} \rightarrow B)$  selon la méthode de Galin, Leclercq et Akkouche [34].
3. Pour chaque nœud et feuille générique  $\mathcal{G}(t)$ , définition de  $\mathcal{P}_{ij}(t)$  par interpolation de  $\mathcal{P}_{A_i}$  et de  $\mathcal{P}_{B_j}$  où  $\mathcal{P}_{A_i}$  et  $\mathcal{P}_{B_j}$  sont des vecteurs de paramètres qui évoluent selon les mouvements  $\mathcal{M}_A$  et  $\mathcal{M}_B$ .
4. Définition du seuil  $\mathcal{E}(t)$  par interpolation des seuils  $T_A$  et  $T_B$ .

#### 4.1.1 Métamorphose de mouvement

Rappelons que l'animation d'un BlobTree est l'ensemble des transformations qui ne modifient pas le type de ses nœuds et de ses feuilles. Aussi, bien que  $A$  et  $B$  soient génériques il est tout de même possible d'établir un graphe de correspondance invariant au cours du temps. Dès lors, nous sommes confrontés aux mêmes problèmes que lors de la métamorphose de deux BlobTrees classiques ; nous aurons donc recours aux mêmes méthodes pour résoudre les étapes 1, 2 et 4. A partir de là, il ne reste plus qu'à définir le BlobTree métamorphosé générique en interpolant les paramètres de définition des nœuds et des feuilles. Nous détaillons ici la partie 3 de l'algorithme.

Comme nous l'avons vu, chaque paramètre intrinsèque de définition du mouvement d'un BlobTree animé est représenté par une spline par morceaux de degrés quelconques. Cependant, dans toute la suite nous continuerons de considérer le cas des Hermite-splines<sup>2</sup> pour le bon rapport qualité-coût qu'elles offrent. Le but de l'interpolation de paramètres est de définir  $\mathcal{P}_C(t)$  selon un mouvement  $\mathcal{M}_C$  interpolant  $\mathcal{M}_A$  et  $\mathcal{M}_B$  entre deux instants définissant l'*animorphose*.

Nous avons choisi de ne pas définir l'interpolation d'un paramètre linéairement afin de garantir la généralité de la méthode en restant dans l'espace stable des splines cubiques. En effet, l'interpolation linéaire de deux fonctions cubiques génère une fonction quartique. Aussi, pour garder un degré constant, nous utilisons des Hermite-splines par morceaux ce qui permet de créer des animorphoses multiples génériques comme nous le verrons plus loin. En pratique, étant donné un intervalle temporel d'animorphose  $[t_{\min}; t_{\max}]$ , il faut tout d'abord déterminer les instants auxquels nous définirons les étapes clés de  $\mathcal{M}_C$ .

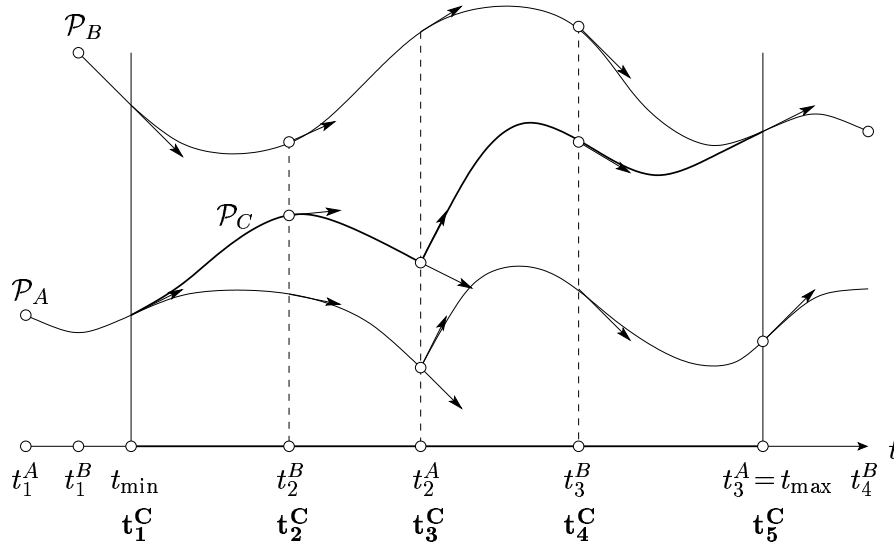


**Figure 3.3.** Découpage temporel régulier pour la définition des étapes clés de  $C$  ; cette discrétisation temporelle fait perdre une caractéristique du mouvement  $\mathcal{M}_A$  pour  $\mathcal{M}_C$ .

Comme l'illustre la figure 3.3, il n'est pas possible d'utiliser un découpage temporel régulier pour définir ces instants de contrôle sous peine de perdre certaines caractéristiques des animations  $\mathcal{M}_A$  et  $\mathcal{M}_B$ . Les

<sup>2</sup>splines de degré 3 définies par les couples valeur-vitesse en deux instants clés successifs.

étapes clés de  $\mathcal{M}_C$  seront donc définies aux instants  $t_{\min}$ ,  $t_{\max}$  et  $\{t_k^C \in ]t_{\min}; t_{\max}[ \}$  choisis parmi les  $\{t_i^A\}$  et  $\{t_j^B\}$  où  $\{t_i^A\}$  et  $\{t_j^B\}$  sont les instants clés respectifs de  $\mathcal{M}_A$  et  $\mathcal{M}_B$ . L'augmentation du nombre d'instants clés inhérente à ce choix est le prix à payer afin de garantir la qualité et le contrôle de l'*animorphose*, mais elle n'a aucune incidence sur les temps de calcul et est négligeable en terme d'espace mémoire puisqu'elle ne concerne que les paramètres de définition de l'animation  $\mathcal{M}$ .



**Figure 3.4.** Cette fois, les étapes clés de  $C$  sont les instants clés de  $A$  ou de  $B$  compris entre  $t_{\min}$  et  $t_{\max}$  : la discontinuité de l'animation  $\mathcal{M}_A$  influence le mouvement  $\mathcal{M}_C$ .

La discrétisation de  $[t_{\min}; t_{\max}]$  étant établie, il faut maintenant définir les étapes clés de  $\mathcal{M}_C$  aux instants  $\{t_k^C\}$ . Pour cela, les instants  $\{t_k^C\}$  sont considérées comme des étapes clés de  $\mathcal{M}_A$  et  $\mathcal{M}_B$ . Le couple valeur-vitesse définissant un paramètre de  $\mathcal{M}_C$  en  $t_k^C$  est alors obtenu par interpolation selon une spline de degré quelconque des couples de ce paramètre pour  $\mathcal{M}_A$  et  $\mathcal{M}_B$  au même instant. Le choix du degré d'interpolation est commun à tous les paramètres et permet de contrôler à un haut niveau la métamorphose de  $(A, \mathcal{M}_A)$  en  $(B, \mathcal{M}_B)$ . Les étapes clés de  $\mathcal{M}_C$  ainsi définies, le mouvement  $\mathcal{M}_C$  est alors celui d'un *BlobTree animé* classique dont l'animation est définie par interpolation à l'aide de splines par morceaux de ses étapes clés (Figure 3.4).

#### 4.1.2 Séquences d'animorphose

Le code d'animation et l'*animorphose* ont été développés et intégrés au modèle du BlobTree. La documentation de l'ensemble du projet et les *animorphoses* que nous présentons sur les pages suivantes sont respectivement visibles sur Internet aux adresses <http://www710.univ-lyon1.fr/~egaline/> et <http://www710.univ-lyon1.fr/~a-barb97/publi/dea/>.

La figure 3.5 présente quelques étapes de l'animation d'un ballon. Cette animation résulte de la métamorphose de deux mouvements : des rebonds verticaux et un roulement horizontal. Il s'agit d'une *animorphose* particulière où les squelettes des objets initial et final sont identiques. Cet exemple illustre bien l'aptitude de l'*animorphose* à définir plus simplement des animations à partir de leurs composantes intrinsèques, offrant ainsi un contrôle plus intuitif.

La figure 3.6 présente des étapes de l'*animorphose* de deux formes plus complexes. L'objet initial est constitué de l'union d'un cône, d'une arête, d'un point et de deux splines ; son mouvement est rectiligne uniforme selon l'horizontale. L'objet final consiste en un mélange d'une cône, d'une arête verticale et de la torsion selon la verticale du mélange d'une croix horizontale avec deux splines ; son mouvement est la composition d'une chute verticale (accélération de la vitesse) à droite de la scène et d'une rotation par rapport à son axe directeur. On peut remarquer que la torsion et la rotation sont de plus en plus rapides au fur et à mesure de la transformation. On peut également noter l'influence des vitesses des objets initial et final aux instants 0 et 1 sur la trajectoire et l'orientation du modèle géométrique intermédiaire.

La figure 3.8 montre la transformation d'un ours marchant en un oiseau en vol, reprenant ainsi la

métamorphose de Plouf et Flap [34] afin d'illustrer le passage des BlobTrees fixes aux *BlobTrees animés*. L'ours marche en ligne droite. Durant deux tiers de la séquence, l'oiseau le survole à une hauteur stable puis il vire sur sa gauche en prenant de l'altitude. Outre la métamorphose des objets, la métamorphose des mouvements apparaît notamment par la transition de l'animation des *bras* en opposition de phase vers des battements d'ailes synchrones. D'autre part, l'interpolation des *jambes* qui sont mises en correspondance avec une unique primitive fantôme en bas du corps de l'oiseau ne suit pas une vitesse linéaire sur tout l'intervalle de travail mais sur une partie uniquement afin d'augmenter la crédibilité de la forme intermédiaire au cours de la transformation.

Il est notable que du fait de la généralité du modèle intermédiaire, l'*animorphose* s'effectue en temps réel puisque la détermination d'une des 950 instances du *BlobTree animorphé* de la figure 3.8 se fait en moyenne en  $553 \mu s$  sur un Athlon à 900 Mhz avec 128 Mo de RAM. Malheureusement, la visualisation en lancer de rayons est coûteuse puisque le rendu d'une image prend en moyenne  $419 s$ . Le décor de la scène est entièrement modélisé avec des BlobTrees classiques.

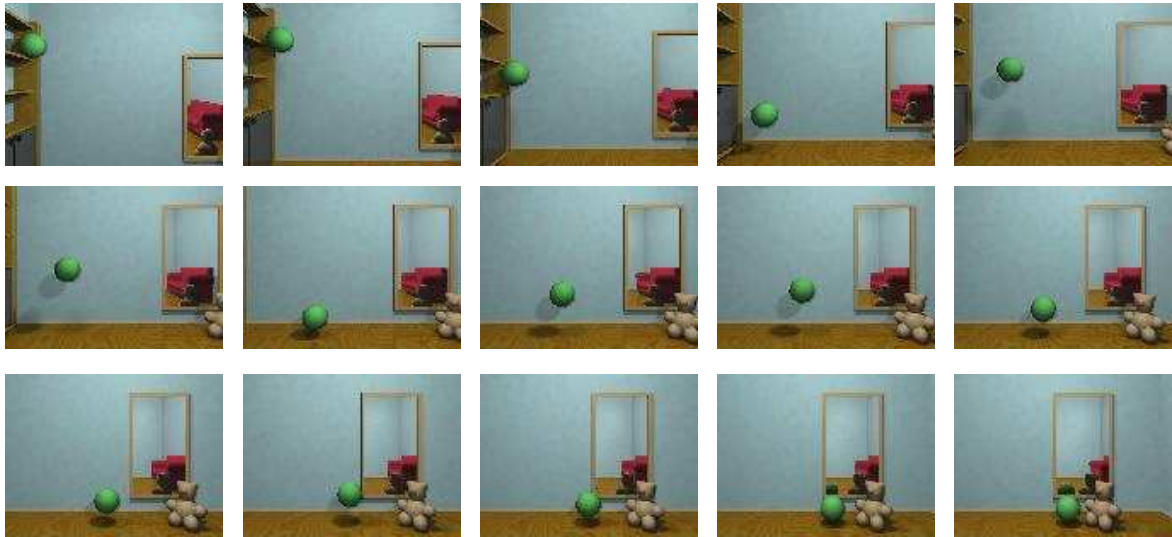
Comme nous le verrons en conclusion, cet exemple prouve qu'il reste beaucoup à faire pour arriver à une visualisation interactive des BlobTrees, qu'ils soient animés ou non.

## 4.2 Animorphose de Bézier

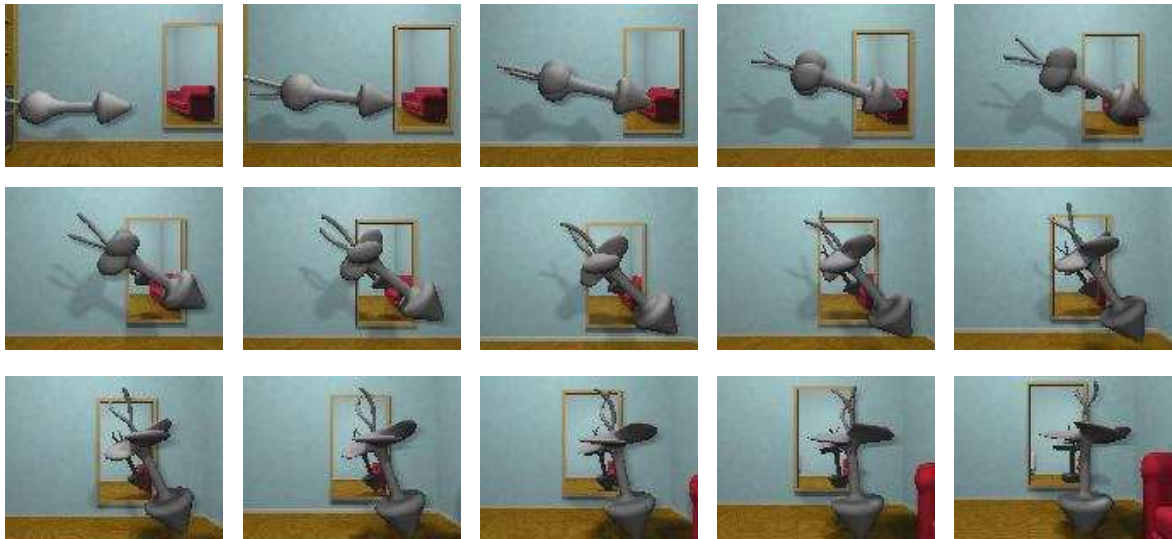
Le fait que  $(C, \mathcal{M}_C)$ , bien que générale, soit un *BlobTree animé* classique permet de définir des *animorphoses* multiples, c'est-à-dire impliquant plus de deux *BlobTrees animés*. L'*animorphose* se fait dans l'espace des *BlobTrees animés*, les points de contrôle des courbes de Bézier étant remplacés par des *BlobTrees animés* de contrôle comme le montre la figure 3.7.

Dans cet exemple, en terme de squelettes, on réalise une *animorphose* entre un cylindre vertical qui tombe et un point qui roule vers un cube qui tourne. L'influence du *BlobTree animé* intermédiaire est maximale pour  $t = 1/2$  même si le point et son mouvement n'apparaissent jamais en entier. Ainsi, le *BlobTree animorphé* cylindre-point qui tombe et roule a pu être réutilisée dans une *animorphose* avec le cube qui tourne. Le squelette résultant est un cylindre-point-cube, c'est-à-dire un cylindre-cube.

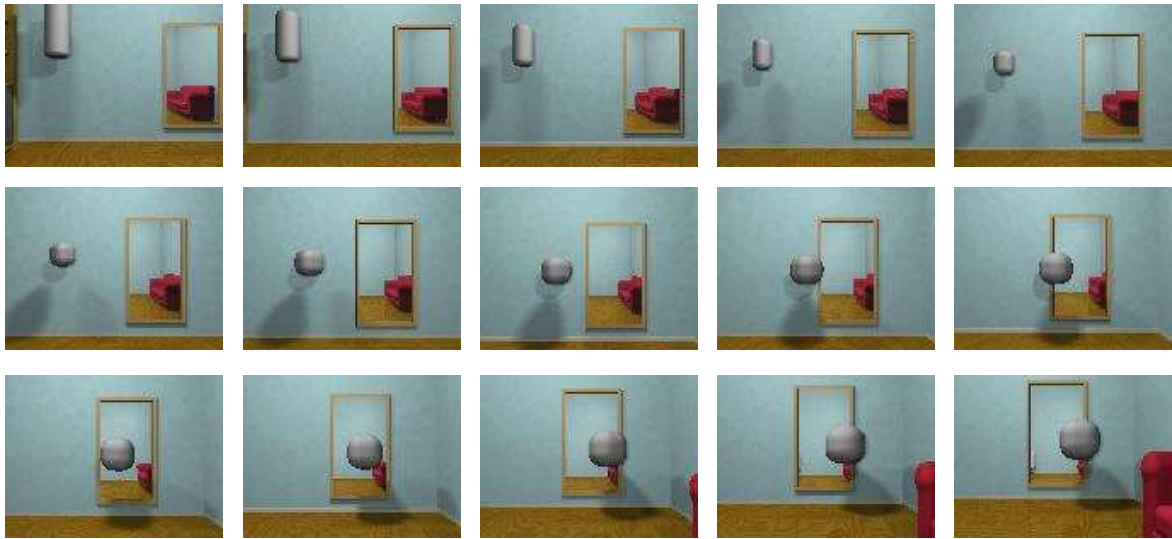
L'*animorphose de Bézier* offre un moyen simple et efficace de contrôle de haut niveau de la transformation. En outre, elle n'a aucune incidence sur le temps de calcul puisqu'elle entraîne la définition d'une unique *BlobTree animorphé* générale, ce qui est effectué en pré-traitement.



**Figure 3.5.** Chute d'un ballon d'éfinie par *animorphose* de deux mouvements. Le résultat est la combinaison d'une animation de rebonds verticaux et d'un roulement horizontal.



**Figure 3.6.** *Animorphose* de deux objets : le premier est en translation de gauche à droite et le second en rotation et translation de haut en bas à droite de l'image.



**Figure 3.7.** Animorphose de Bézier entre un cylindre qui tombe à gauche de l'image, une sphère qui roule sur le sol de gauche à droite et un cube en rotation autour de la verticale en bas à droite de l'image.



**Figure 3.8.** Plouf et Flap [34] sont maintenant animés. **Attention, il faut lire de droite à gauche !**

## Conclusions et Perspectives

Nous avons démontré la faisabilité du mixage d'animation et de métamorphose d'objets complexes en proposant une extension du modèle du BlobTree. Nos contributions à ce modèle sont triples :

- Nous avons étendu la variété des primitives à squelettes intervenant dans la modélisation de formes complexes, en ajoutant en particulier des éléments à squelettes splines et volumiques, ce qui simplifie leur définition et accroît leur qualité visuelle. Nous avons également défini la métamorphose de ces nouvelles primitives.
- Nous avons proposé un système d'animation cinématique reposant sur l'interpolation des paramètres de définition des composantes du *BlobTree animé* selon des splines par morceaux de degrés quelconques. Ce choix satisfait notre exigence de contrôle intuitif et précis lors de la définition d'une animation et de corrections de trajectoire éventuelles.
- Enfin, Nous avons proposé une méthode de métamorphose du *BlobTree animé*. Nous définissons un modèle intermédiaire générique en garantissant la stabilité du degré des splines qui déterminent l'interpolation des paramètres entre leurs étapes clés. La cohérence de la forme intermédiaire est assurée par l'utilisation de repères locaux dont l'orientation définie à l'aide de quaternions permet une interpolation douce sans perte de degré de liberté. En outre, la généralité du modèle intermédiaire autorise la définition d'*animorphoses de Bézier* dont les éléments de contrôle sont des *BlobTrees animés*, ce qui permet un contrôle de haut niveau aisé lors de la transformation.

L'*animorphose* génère des transformations esthétiquement convaincantes entre des objets animés complexes ce qui accroît le champ des effets spéciaux possibles. En outre, elle apparaît comme une méthode simple et efficace de définition d'animation par interpolation des différentes composantes caractérisant le mouvement.

Pour autant, de nombreuses améliorations sont souhaitables afin de favoriser le développement de l'utilisation des surfaces implicites pour la création graphique. Ainsi, la gestion de textures différentes pour les primitives du BlobTree est indispensable pour la définition d'objets complexes de qualité. Il faudra donc définir des opérateurs de combinaison de textures comme cela existe pour les fonctions potentiel.

Une évolution du système d'animation vers un système cinématique à contrôle dynamique par adjonction d'un superviseur tirant parti de la dynamique inverse augmenterait le réalisme physique des animations. De même, la gestion des collisions et inter-pénétrations d'objets déchargerait l'animateur de certaines corrections de trajectoire lorsqu'il le désire.

Afin de visualiser et d'éditer les BlobTrees en temps interactif, des méthodes de maillage spécifiques sont nécessaires. De même, des techniques de lancer de rayon accélérées sont souhaitables pour la visualisation d'animations d'objets complexes [103]. De nombreux travaux restent à faire dans ces domaines. Dans le même but, la définition d'un BlobTree à niveaux de détail permettrait des gains de temps considérables lors du rendu de scènes complexes. Elle pourrait être réalisée à l'aide d'un arbre à niveaux de détail [14] et tirer parti de la métamorphose pour générer des transitions douces entre les différents niveaux de détail.



# Bibliographie et Netographie

## Bibliographie

- [1] Samir Akkouche, Eric Galin. Adaptative Implicit Surface Polygonization using Marching Triangles. *Computer Graphics Forum*. **20**(2). 2001.
- [2] Alan H. Barr. Global and Local Deformations of Solid Primitives. *Computer Graphics (Siggraph'84 Proceedings)*. **18**(3):21-30. 1984.
- [3] Dominique Bechmann. Space Deformation Models Survey. *Computer & Graphics*. **18**(4):571-586. 1994.
- [4] Thaddeus Beier, Shawn Neely. Feature-Based Image Metamorphosis. *Computer Graphics (Siggraph'92)*. **26**:35-42. 1992.
- [5] W.E. Bethel, S.P. Uselton. Shape Distorsion in Computer-Assisted Keyframe Animation. *Computer Animation'89, State-of-the-art in Computer Animation, Computer Graphics International, Springer, Berlin Heidelberg New-York*. 215-224. 1989.
- [6] A. Bharatkumar, K. Daigle, M. Pandey, Q. Cai, J. Aggarwal. Lower limb kinematics of human walking with the medial axis transformation. *In IEEE Workshop on Non-Rigid Motion*. 70-76. 1994.
- [7] Carole Blanc, Christophe Schlick. Ratioquadratics : an Alternative Model for Superquadratics. *The Visual Computer*. **12**:420-428. 1996.
- [8] Carole Blanc, Christophe Schlick. Extended Field Functions for Soft Objects. *Proceedings of Implicit Surfaces'95*. 21-32. 1995.
- [9] Jules Bloomenthal with Chandrajit Bajaj, Jim Blinn, Marie-Paule Cani-Gascuel, Alyn Rockwood, Brian Wyvill and Geoff Wyvill. Introduction to Implicit Surfaces. *Morgan Kaufmann Publishers Inc, San Francisco, California*. 1997.
- [10] Jules Bloomenthal, Brian Wyvill. Interactive Techniques for Implicit Modeling. *Computer Graphics*. **24**(2):109-116. 1990.
- [11] Jules Bloomenthal. Polygonization of Implicit Surfaces. *Computer Aided Geometric Design*. **5**:341-355. 1988.
- [12] David E. Breen, Ross T. Whitaker. A Level-Set Approach for the Metamorphosis of Solid Models. *IEEE Transactions on Visualization and Computer Graphics*. 2001.
- [13] David E. Breen, Sean Maunch, Ross T. Whitaker, Jia Mao. 3D Metamorphosis Between Different Types of Geometric Models. *Eurographics'2001*. **20**(3). 2001.
- [14] Marie-Paule Cani, Samuel Hornus. Subdivision-Curve Primitives : a New Solution for Interactive Implicit Modeling. *Shape Modelling International*. 2001.

- [15] Marie-Paule Cani-Gascuel, Mathieu Desbrun. Animation of Deformable Models using Implicit Surfaces. *IEEE Transaction of Visualization and Computer Graphics*. **3**(1):39-50. 1997.
- [16] M. Chen, M.W. Jones, P. Townsend. Methods for Volume Metamorphosis. *European Workshop on Combined Real and Synthetic Image Processing for Broadcast and Video Production*. 1994.
- [17] Benoit Crespın, Pascal Guitton, Christophe Schlick. Efficient and Accurate Tessellation of Implicit Sweep Objects. *Constructive Solid Geometry'98*. 1998.
- [18] Benoit Crespın, Carole Blanc, Christophe Schlick. Implicit Sweep Objects. *Eurographics'96*. **15**(3):165-174. 1996.
- [19] Daniel Cohen-Or, David Levin, Amira Solomovici. Three-Dimensional Distance Field Metamorphosis. *ACM Transactions on Graphics*. **17**(2):116-141. 1998.
- [20] Wagner Toledo Corrêa, Robert J. Jensen, Craig E. Thayer, Adam Finkelstein. Texture Mapping for Cel Animation. *Siggraph'98 Proceedings*. 435-446. 1998.
- [21] Gilles Debunne, Mathieu Desbrun, Alan Barr, Marie-Paule Cani. Interactive Multiresolution Animation of Deformable Models. *10th Eurographics Workshop on Computer Animation and Simulation*. 1999.
- [22] Douglas DeCarlo, Jean Gallier. Topological Evolution of Surfaces. *Graphics Interface'96*. 194-203. 1996.
- [23] Philippe Decaudin, Andr e Gagalowicz. Fusion of 3D Shapes. *The 5th Eurographics Workshop on Animation and Simulation*. 1-14. 1994.
- [24] H. Delingette, Y. Watanabe, Yasuhito Suenaga. Simplex Based Animation. *Computer Animation'93, Models and Techniques in Computer Animation, Computers Graphics International Geneva*. 11-28. 1993.
- [25] Erik Demaine, Martin Demaine, Anna Lubiw, Joseph O'Rourke, Irena Paschenko. Metamorphosis of the Cube. *ACM Symposium on Computational Geometry*. 1999.
- [26] Tony DeRose, Michael Kass, Tien Truong. Subdivision Surfaces in Character Animation. *Computer Graphics Proceedings, Annual Conference Series (Siggraph'98)*. 85-94. 1998.
- [27] Mathieu Desbrun. Modeling and Animating Highly Deformable Objects in Computer Graphics. *Ph.D. Thesis, Institut National Polytechnique de Grenoble, Grenoble, France*. part 2, chapters 4 & 5, 61-97. 1998.
- [28] Mathieu Desbrun, Marie-Paule Cani-Gascuel. Smoothed Particles : a New Paradigm for Animating Highly Deformable Bodies. *6th Eurographics Workshop on Animation and Simulation*. 1996.
- [29] Mathieu Desbrun, Marie-Paule Gascuel. Animating Soft Substances with Implicit Surfaces. *Siggraph 95 Conference Proceedings, Annual Conference Series*. 287-290. 1995.
- [30] Mathieu Desbrun, Nicolas Tsingos, Marie-Paule Gascuel. Adaptive Sampling of Implicit Surfaces for Interactive Modeling and Animation. *Implicit Surfaces '95 Proceedings*. 171-185. 1995.
- [31] Mathieu Desbrun, Marie-Paule Gascuel. Highly Deformable Material for Animation and Collision Processing. *5th Eurographics Workshop on Animation and Simulation*. 1994.
- [32] Franois Faure, Gilles Debunne, Marie-Paule Cani-Gascuel, Franck Multon. Dynamic Analysis of Human Walking. *8th Eurographics Workshop on Computer Animation and Simulation*. 1997.
- [33] Eric Ferley, Marie-Paule Cani-Gascuel, Dominique Attali. Skeletal Reconstruction of Branching Shapes. *Computer Graphics Forum*. **16**(5). 1997.
- [34] Eric Galin, Antoine Leclercq, Samir Akkouche. Morphing the Blob-Tree. *Computer Graphics Forum*. **19**(4):257-270. 2000.
- [35] Eric Galin, Samir Akkouche. Incremental Polygonization of Implicit Surfaces. *Graphical Models*. **62**:19-39. 2000.

- [36] Eric Galin. M'etamorphose et Visualisation de Blobs à Squelettes. *Ph.D. Thesis, University Claude Bernard Lyon 1, Lyon, France.* 1997.
- [37] Eric Galin, Samir Akkouche. Soft Objects Metamorphosis Based on Minkowski Sums. *Proceedings of Eurographics'96.* **15**(3):143-153. 1996.
- [38] Eric Galin, Samir Akkouche. Shape Constrained Blob Metamorphosis. *Proceedings of Implicit Surfaces'96.* 9-23. 1996.
- [39] Jean-Dominique Gascuel, Marie-Paule Cani-Gascuel, Mathieu Desbrun, E. Leroy, C. Mignon. Simulating Landslides for Natural Disaster Prevention. *In 9th Eurographics Workshop on Computer Animation and Simulation.* 1998.
- [40] Jean-Dominique Gascuel, Marie-Paule Gascuel. Displacements Constraints : a New Method for Interactive Dynamic Animation of Articulated Bodies. *Eurographics'92 Proceedings.* 1992.
- [41] E. Goldstein, C. Gotsman. Polygon Morphing Using a Multiresolution Representation. *Proceedings of Graphics Interface'95.* 247-254. 1995.
- [42] Arthur Gregory, Andrei State, Ming C. Lin, Dinesh Manocha, Mark A. Livingston. Feature-based Surface Decomposition for Correspondence and Morphing between Polyhedra. *Department of Computer Science, University of North Carolina, Technical Report TR98-014.* 1998
- [43] Igor Guskov, Wim Sweldens, Peter Schröder. Multiresolution Signal Processing for Meshes. *Computer Graphics (Siggraph'99 Proceedings).* 325-334. 1999.
- [44] John C. Hart. Shere Tracing : a Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces. *The Visual Computer.* **12**(10):527-545. 1996.
- [45] John C. Hart. Ray Tracing Implicit Surfaces. *Siggraph'93 Course Notes 25 : Modelling, Visualizing and Animating Implicit Surfaces.* 1993.
- [46] Taosong He, Sidney Wang, Arie Kaufman. Wavelet-Based Volume Morphing. *in Bergeron RD, Kaufman AE (eds) Proceedings of Visualization'94, IEEE Computer Society Press, Washington DC.* 85-92. 1994.
- [47] Jessica K. Hodgins, Nancy S. Pollard. Physically Realistic Morphing. *Technical Report GIT-GVU-97-02.* 1997
- [48] T.M. Hong, N. Magnetat-Thalmann, Daniel Thalmann. A General Algorithm for 3D Shape Interpolation in a Facet-Based Representation. *Graphics Interface'88, Edmonton, Canada, Canadian Information Processing Society, Palo-Alto.* 229-235. 1988.
- [49] John F. Hughes. Scheduled Fourier Volume Morphing. *Computer Graphics (Siggraph'92).* **26**:43-46. 1992.
- [50] Henry Johan, Yuichi Koiso, Tomoyuki Nishita. Morphing using Curves and Shape Interpolation Techniques. *Proceedings of the Pacific Graphics 2000.* 2000.
- [51] Lee Joo-Haeng, Kim Myoung-Soo. Pseudo Dynamic Keyframe Animation with Motion Blending on the Configuration Space of a Moving Mechanism. *in Computer Graphics and Applications, S.Y. Shin and T.L. Kunii (Eds.), World Scientific.* 118-132. 1995.
- [52] Marcelo E. Kallmann, Antonio A. F. Oliveira. Homeomorphisms and Metamorphosis of Polyhedral Models Using Fields of Directions Defined on Triangulations. *Journal of the Brazilian Computer Society ISSN 0104-6500.* **3**(3). 1997.
- [53] Devendra Kalra, Alan H. Barr. Guaranteed Ray Intersections with Implicit Surfaces. *Computer Graphics.* **23**(3):297-306. 1989.
- [54] T. Kanai, H. Suzuki, F. Kimura. 3D Geometric Metamorphosis Based on Harmonic Maps. *Visual Computer.* **14**:166-176. 1998.
- [55] Anil Kaul, Jarek Rossignac. Solid Interpolating Deformations : Construction and Animation of PIPS *Eurographics'91.* 493-505. 1991.

- [56] James R. Kent, Wayne E. Carlson, Richard E. Parent. Shape Transformation for polyhedral Objects. *Computer Graphics (Siggraph'92)*. **26**:47-54. 1992.
- [57] James R. Kent, Richard E. Parent, Wayne E. Carlson. Establishing Correspondences by Topological Merging : a New Approach to 3D Shape Transformation. *Graphics Interface'91*. 271-278. 1991.
- [58] M.J. Kim, M.S. Kim, S. Shin. A General Construction Scheme for Unit Quaternion Curves with Simple High Order Derivatives. *Computer Graphics (Siggraph'95 Proceedings)*. 369-376. 1995.
- [59] Leif P. Kobbelt, Thilo Bareuther, Hans-Peter Seidel. Multiresolution Shape Deformation for Meshes with Dynamic Vertex Connectivity. *Eurographics'2000*. **19**(3). 2000.
- [60] Leif Kobbelt, Swen Campagna, Jens Vorsatz, Hans-Peter Seidel. Interactive Multiresolution Modeling on Arbitrary Meshes. *Computer Graphics (Siggraph'98 Proceedings)*. 105-114. 1998.
- [61] Venkat Krishnamurthy, Marc Levoy. Fitting Smooth Surfaces to Dense Polygon Meshes. *Computer Graphics (Siggraph'96 Proceedings)*. 313-324. 1996.
- [62] Alexis Lamouret, Marie-Paule Gascuel. Scripting Interactive Physically-Based Motions with Relative Paths and Synchronization. *Computer Graphics Forum*. **15**(1). 1996.
- [63] Alexis Lamouret, Marie-Paule Gascuel, Jean-Dominique Gascuel. Combining Physically-Based Simulation of Colliding Objects with Trajectory Control. *The Journal of Visualization and Computer Animation*. **6**(2):71-90. 1995.
- [64] Alexis Lamouret. Animation par modèles g'én'érateurs : contrôle du mouvement. *Ph.D. Thesis, University Joseph Fourier, Grenoble, France*. chapter 3, 89-107. 1995.
- [65] François Lazarus, Anne Verroust. Three-dimensional metamorphosis : a survey. *The Visual Computer*. **14**:373-389. 1998.
- [66] François Lazarus, Anne Verroust. Metamorphosis of Cylinder-like Objects. *Visualization Computer Animation*. **8**:131-146. 1997.
- [67] François Lazarus, Anne Verroust. Décomposition cylindrique de polyèdre et courbe squelette. *Revue Internationale de CFAO et d'Infographie*. **11**:363-377. 1996.
- [68] Antoine Leclercq, Samir Akkouche, Eric Galin. Mixing Triangle Meshes and Implicit Surfaces in Character Animation. *Eurographics Workshop 2001 on Computer Animation and Simulation*. 2001.
- [69] Jehee Lee, Sung Yong Shin. A Hierarchical Approach to Interactive Motion Editing for Human-like Figures. *Proceedings of Siggraph'99*. 39-48. 1999.
- [70] Joo-Haeng Lee, Myung-Soo Kim. Pseudo Dynamic Keyframe Animation with Motion Blending on the Configuration Space of a Moving Mechanism. in *Computer Graphics and Applications (Pacific Graphics'95 proceedings)*. 118-132. 1995.
- [71] Apostolos Lierios, Chase D. Garfinkle, Marc Levoy. Feature-Based Volume Metamorphosis. *Computer Graphics (Siggraph'95)*. **29**:449-464. 1995.
- [72] William E. Lorensen, Harvey E. Cline. Marching Cubes : a High Resolution 3D Surface Construction Algorithm *Computer Graphics (Siggraph'87 Proceedings)*. **21**(4):163-169. 1987.
- [73] Ron Mac Cracken, Kenneth I. Joy. Free-Form Deformations with Lattices of Arbitrary Topology. *Computer Graphics Proceedings, Annual Conference Series*. 181-188. 1996.
- [74] Oleg Mazarak, Claude Martins, Johan Amanatides. Animating Exploding Objects. *Graphics Interface'99*. 1999.
- [75] Don P. Mitchell. Robust Ray Intersection with Interval Arithmetic. *Proceedings of Graphics Interface'90, Morgan Kaufman*. 68-74. 1990.
- [76] Tomas Möller, Eric Haines. Real-Time Rendering. *A K Peters, Natick, Massachusetts*. 1999.
- [77] Claudio Montani, Riccardo Scateni, Roberto Scopigno. A Modified Look-up Table for Implicit Disambiguation of Marching Cubes. *The Visual Computer*. **10**:353-355. 1994.

- [78] Franck Multon, Laure France, Marie-Paule Cani-Gascuel, Gilles Debunne. Computer Animation of Human Walking : a Survey. *The Journal of Visualization and Computer Animation*. **6**(2). 1999.
- [79] S. Murakami, H. IchiHara. On a 3D Display Method by Metaball Technique. *Journal of papers at Electronics Communication Conference'87*. **J70-D**(8):1607-1615. 1987.
- [80] Fabrice Neyret, Marie-Paule Cani. Pattern-Based Texturing Revisited. *Computer Graphics (Proceedings of Siggraph'99)*. 235-242. 1999.
- [81] Fabrice Neyret. Modeling Animating and Rendering Complex Scenes using Volumetric Textures. *IEEE Transaction on Visualization and Computer Graphics*. **4**(1):55-70. 1998.
- [82] Agata Opalach, Steve Maddock. High Level Control of Implicit Surfaces for Character Animation. *Proceedings 1st International Eurographics Workshop on Implicit Surfaces*. 1995.
- [83] Richard E. Parent. Shape Transformation by Boundary Representation Interpolation : a recursive approach to establishing Face Correspondences. *Journal of Visualization in Computer Animation*. **3**:219-239. 1992.
- [84] Alexander Pasko, Vladimir Savchenko. Constructing Functionally Defined Surfaces. *Proceedings of Implicit Surfaces'95*. 97-106. 1995.
- [85] Frederic Pighin, Joel Auslander, Dani Lischinski, David H. Salesin, Richard Szeliski. Realistic Facial Animation Using Image-Based 3D Morphing. *Technical Report UW-CSE-97-01-03*. 1997.
- [86] Ravi Ramamoorthi, Cindy Ball, Alan H. Barr. Dynamic Splines with Constraints for Animation. *Technical Report CS-TR-97-03, California Institute of Technology*. 1997.
- [87] K. Rohr. Human Movement Analysis Based On Explicit Motion Models. *Motion based recognition, M. Shah and R. Jain (Eds.), Kluwer academic publishers*. chapter 8, 171-198. 1997.
- [88] Jarek Rossignac, Anil Kaul. AGRELS and BIBs : Metamorphosis as a B´ezier Curve in the Space of Polyhedra. *Eurographics'94*. 179-184. 1994.
- [89] Ken Schoemake. Animating Rotation with Quaternion Curves. *Computer Graphics (Siggraph'85 Proceedings)*. **19**(3):245-254. 1985.
- [90] Thomas W. Sederberg, Scott R. Parry. Free-Form Deformation of Solid Geometric Models. *Siggraph'86*. **20**(4):151-160. 1986.
- [91] Y.M. Sun, W. Wang, F.Y. Chin Interpolating Polyhedral Models Using Intrinsic Shape Parameters. *Visualization Computer Animation*. **8**:81-96. 1997.
- [92] Greg Turk, James F. O'Brien. Variational Implicit Surfaces. *Technical Report GIT-GVU-99-15, Georgia Institute of Technology*. 1999.
- [93] Greg Turk, James F. O'Brien. Shape Transformation Using Variational Implicit Functions. *Proceedings of Siggraph'99*. 335-342. 1999.
- [94] Anne Verroust, Francis Lazarus. Extracting Skeletal Curves from 3D Scattered Data. *The Visual Computer*. **16**(1):15-25. 2000.
- [95] R. Westermann, Leif Kobbelt, T. Ertl. Real-Time Exploration of Regular Volume Data by Adaptive Reconstruction of Isosurfaces. *The Visual Computer*. **15**: 100-111. 1999.
- [96] Andrew Witkin, Zoran Popovi´c. Motion Warping. *Computer Graphics (Siggraph'95 Proceedings)*. 1995.
- [97] Andrew Woo, Pierre Poulin, Alain Fournier. A Survey of Shadow Algorithms. *IEEE Computer Graphics and Applications*. **10**(6):13-32. 1990.
- [98] Brian Wyvill, Andrew Guy, Eric Galin. Extending the CSG Tree (Warping, Blending, and Boolean Operations in an Implicit Surface Modeling System). *Computer Graphics Forum*. **18**(2):149-158. 1999.
- [99] Brian Wyvill. Metamorphosis of Implicit Surfaces. *Siggraph'93, Course Notes*. **25**. 1993.

- [100] Brian Wyvill, Jules Bloomenthal, Thaddeus Beier, Jim Blinn, Alyn Rockwood, Geoff Wyvill. Modeling and Animating with Implicit Surfaces. *Siggraph Course Notes* **23**. 1990.
- [101] Geoff Wyvill, Andrew Trotman. Ray-Tracing Soft Objects. *Proceedings of Computer Graphics International'90*. 469-476. 1990.
- [102] Brian Wyvill, Craig Mc Pheeters, Geoff Wyvill. Data Structure for Soft Objects. *The Visual Computer*. **2**(4):227-234. 1986.
- [103] Roni Yagel, Zhouhong Shi. Accelerating Volume Animation by Space-Leaping. *Proceedings of Visualization'93*. 62-69. 1993.

## **Netographie**

- [104] Infografica: [http://www.reyes-infografica.com/meta\\_p.php](http://www.reyes-infografica.com/meta_p.php)



## Abstract

Although several studies on animation and metamorphosis have been extensively discussed in literature, mixing both animation and metamorphosis has never been proposed as far as we know. In this paper, we define *animorphosis*, i.e. blending of animation and metamorphosis, for some kind of implicit surfaces. Our goal is to integrate the *animorphosis* concept in the BlobTree model. This modeling system defines an implicit surface as a hierarchical construction tree.

Our includes the following contributions : we have developped new complex skeletal primitives (volumes, spline curves) in order to enhance modelization possibilities. We present a keyframe animation system that may be enhanced to include dynamic simulation however. The animation of the BlobTree is based on interpolation of parameters characterizing the BlobTree's components between two frames. Eventually, animorphosis is defined by a generic inbetweening model and by the interpolation of its characteristic.

Animation control, such as trajectory correction, is of primary importance to obtain convincing special effects. This control is simply realized by moving, inserting, or deleting control points in the sequence interpolation. High level control is also provided thanks to *Bézier-like animorphosis* which generalizes animorphosis between several control shapes.

## Résumé

Bien que de nombreuses études sur l'animation et la métamorphose existent dans la littérature, le mélange d'animation et de métamorphose n'a jamais été abordé à notre connaissance. Nous nous proposons ici de définir l'*animorphose*, c'est-à-dire le mixage d'animation et de métamorphose, pour certaines catégories de surfaces implicites. Notre objectif est d'intégrer le concept d'*animorphose* au BlobTree. Ce modèle définit une surface implicite à l'aide d'un arbre de construction hiérarchique.

Dans ce cadre, notre contribution est triple. Nous avons développé de nouvelles primitives à squelettes complexes (volumes, courbes splines) afin d'accroître les possibilités de modélisation. Nous présentons un système d'animation cinématique pouvant être étendu à un système à validation dynamique. Il repose sur l'interpolation entre les instants clefs des paramètres caractérisant les composantes du BlobTree. Enfin, l'*animorphose* est définie par un modèle intermédiaire générique et l'interpolation des paramètres de définition des composantes apparues dans le graphe de correspondance défini par l'utilisateur.

Le contrôle de l'animation, comme les corrections de trajectoire, élément primordial nécessaire à l'obtention d'effets spéciaux convaincant, est réalisé simplement par le déplacement, l'insertion ou la suppression de points de contrôle dans l'interpolation. Un contrôle de plus haut niveau est présente grâce à l'*animorphose de Bézier* qui généralise l'*animorphose* entre plusieurs formes de contrôle.