Reconstruction 3D d'objets par une representation fonctionnelle

Pierre-Alain Fayolle

DOCTORAT

DE

L'UNIVERSITÉ D'ORLÉANS

Discipline : **Informatique**

PAR

Pierre-Alain FAYOLLE

Reconstruction 3D d'objets
par une représentation fonctionnelle

Soutenue le : **17 Décembre 2007**

MEMBRES DU JURY

- Sébastien Limet	Professeur	Université d'Orléans	**Président**
- Atilla Baskurt	Professeur	INSA de Lyon	**Rapporteurs**
- Luc Brun	Professeur	ENSICAEN	
- Patrick Bourdot	Chargé de recherche CNRS	Université Paris XI	**Examinateurs**
- Christophe Rosenberger	Professeur	ENSICAEN	
- Christian Toinard	Professeur	Université d'Orléans	

Résumé

NOUS nous sommes essentiellement intéressés à la modélisation d'objets volumétriques par des champs de distance scalaire. La distance Euclidienne d'un point a un ensemble de points représentant la frontière d'un solide, correspond à la plus petite distance (définie à partir de la norme Euclidienne) entre ce point et n'importe quel point de l'ensemble. La représentation du solide par la distance à la surface du solide est une méthode concise mais relativement puissante pour définir et manipuler des solides. Dans ce cadre, nous nous sommes intéressés à la modélisation constructive de solides, et à la façon d'implémenter les opérations ensemblistes par des fonctions afin de garantir une bonne approximation de la distance ainsi que certaines propriétés de différentiabilité, nécessaire pour plusieurs classes d'opérations ou applications sur les solides. Nous avons construit différents types de fonctions implémentant les principales opérations ensemblistes (union, intersection, différence). Ces fonctions peuvent être ensuite appliquées à des primitives, définies par la distance à la surface de la primitive, afin de construire récursivement des solides complexes, définies eux-mêmes par une approximation à la distance du solide. Ces fonctions correspondent en fait à une certaine classe de R-fonctions, obtenues en lissant les points critiques des fonctions min/max (qui sont elles mêmes des R-fonctions). Ces fonctions sont appelées Signed Approximate Real Distance Functions (SARDF).

Le cadre SARDF, constitue des fonctions décrites ci-dessus et de primitives définies par la fonction distance, a été utilisé pour la modélisation hétérogène de solides. La distance, ou son approximation, à la surface du solide ou des matériaux internes est utilisée comme un paramètre pour modéliser la distribution des matériaux à l'intérieur du solide. Le cadre SARDF a principalement été implémenté comme une extension de l'interpréteur d'HyperFun et à l'intérieur de l'applet Java d'HyperFun. La modélisation constructive de solides possède de nombreux avantages qui en font un outil puissant pour la modélisation de solides. Néanmoins, la définition constructive de solides peut être fastidieuse et répétitive. Nous avons étudié différents aspects pour l'automatiser.

Dans un premier temps, nous avons introduit la notion de modèles template, et proposé différents algorithmes pour optimiser la forme d'un template à différentes instances correspondant à des nuages de points, sur ou aux alentours de la surface du solide. L'idée des templates vient de l'observation que les solides traditionnellement modélisés par ordinateur peuvent être regroupés en différentes classes possédant

des caractéristiques communes. Par exemple, différents vases peuvent avoir une forme commune. Cette forme générale est modélisée une seule fois, et différents paramètres gouvernant les caractéristiques de la forme sont extraits. Ces paramètres sont ensuite optimisés à l'aide d'une combinaison de méta-heuristique comme le recuit simulé ou les algorithmes génétiques avec des méthodes directes du type Newton ou Levenberg-Marquardt. L'utilisation du cadre SARDF pour la définition du modèle template est préférable, car donne de meilleurs résultats avec les algorithmes d'optimisation. Nous pouvons maintenant nous demander comment le modèle template est obtenu. Une première solution est d'utiliser les services d'un artiste. Néanmoins, nous pouvons aussi réfléchir pour automatiser ce processus. Nous avons essentiellement étudié deux aspects pour répondre à cette question : la première est l'utilisation de la programmation génétique pour former un modèle constructif à partir d'un nuage de points. La deuxième solution consiste à partir d'un nuage de points segmentés et une liste de primitives optimisés à ce nuage de points segmenté, et d'utiliser un algorithme génétique pour déterminer l'ordre et le type d'opérations qui peuvent être appliquées à ces primitives. Ces deux solutions ont été implémentées et leurs résultats discutés.

Mots clés : Modélisation d'objets 3D, optimisation, nuage de points.

Remerciements

Je remercie les Professeurs Atilla Baskurt et Luc Brun d'avoir été les rapporteurs de cette thèse. Je tiens à remercier les examinateurs de ma thèse de doctorat à savoir Patrick Bourdot et Sébastien Limet.

Je tiens à remercier mes directeurs de thèse les professeurs Christian Toinard et Christophe Rosenberger pour leur soutien durant la durée de mes recherches. Je tiens à remercier aussi le professeur Alexander Pasko pour avoir discuter de nombreuses idées et pistes de recherche avec moi. Je remercie le professeur Nikolay Mirenkov qui m'a accueilli dans son laboratoire au Japon ainsi que mes nombreux collègues : Yuichiro Goto, Benjamin Schmitt, le professeur Lothar Schmitt et Carl Vilbrandt.

Je remercie les membres du laboratoire qui m'a accueilli à l'université d'Aizu-Wakamatsu, en particulier le professeur Rentaro Yoshioka et Yutaka Watanobe pour leur aide et les membres du laboratoire d'Informatique Fondamentale d'Orléans.

Je suis reconnaissant au ministère Japonais de l'éducation, de la culture, des sports et des sciences et technologie ainsi qu'au conseil général du Cher pour les bourses qui m'ont aidé à financer mes recherches.

Je voudrais remercier mes amis : la famille Aotsu, la famille Takeda, la famille Kobayashi, Philippe et son épouse Keiko, Hiroko-san, Sakae-san, Mr Izumi, Cyprien, Francois, Philippe et Vincent.

Finalement, je remercie ma femme Asuka, et ma famille en France et au Japon.

Table des matières

CHAPITRE 1

Introduction

La modélisation d'objets, de leurs propriétés et relations est un sujet de recherche important en informatique. Dans ce manuscrit, le terme "objet" fait référence à un ensemble de points géométriques, et nous nous intéressons plus particulièrement à leur modélisation et construction à partir de champs scalaires de distance. Ces "objets" ne sont pas restreints à des courbes en 2 dimensions et surfaces en 3 dimensions mais aussi à leur intérieur et leurs propriétés internes (couleur, matière, propriété physique et autres) ; l'utilisation de champs scalaires basés sur la notion de distance peut naturellement définir de tels objets et sert à décrire ces propriétés internes. Nous utilisons le terme objets volumétriques pour les référencer.

La modélisation et visualisation d'objets volumétriques a de multiples applications comme : la préservation d'héritage culturelle, la visualisation scientifique, les simulations physiques ou chimiques, l'ingénierie mécanique, entre autres.

De nombreux modèles mathématiques ont été développés pour la modélisation et la construction de solides ou objets volumétriques par ordinateur ; chacun ayant ses avantages et inconvénients. Dans ce manuscrit, nous étudions la représentation d'objets volumétriques définis par des champs scalaires continus de distance comme décrit dans la théorie de la représentation fonctionnelle [52]. En effet, dans le modèle de la représentation fonctionnelle (FRep est un acronyme pour Function Representation), un objet volumétrique (solide) est défini par une fonction réelle et continue. Le modèle FRep généralise les surfaces implicites, la géométrie constructive solide (CSG) et d'autres techniques de modélisation de solides et de formes dans un "framework" unifié.

Dans ce travail, nous nous limitons à la modélisation constructive et la création de formes et solides définies par la distance Euclidienne signée à la surface du solide (ou son approximation). La modélisation constructive d'objets est une méthode élégante pour construire des objets en utilisant comme type de données un arbre, dans lequel les noeuds sont des opérations géométriques et les feuilles des primitives géométriques. Cet arbre, appelé dans le reste de ce document un arbre constructif, contient l'information relative à la structure de l'objet, les opérations utilisées pour sa construction ainsi que la sémantique de l'objet. La fonction représentant la distance

Euclidienne signée définie un objet solide en associant à chaque point de l'espace la plus petite distance Euclidienne entre ce point et tout point appartenant à la surface du solide. Le signe est utilisé pour distinguer l'intérieur et l'extérieur du solide.

Les champs scalaires définis à partir de la distance Euclidienne ont l'avantage de définir naturellement et simplement un objet volumique et de simplifier la modélisation d'objets. L'utilisation de la distance Euclidienne fournit un paramètre qui permet de définir des contraintes lors de la modélisation mais permet aussi de définir des propriétés internes de l'objet (encore appelés attributs).

Le travail présenté dans ce manuscrit poursuit différents objectifs :

- la définition de nouvelles expressions pour les opérations ensemblistes utilisées en géométrie constructive : intersection, union et différence. Ces opérations sont définis en termes de fonctions appliquées à des fonctions, définissant les objets géométriques qui vont être combinés. Les fonctions définies ont pour but de proposer une meilleure approximation de la distance que les R-fonctions [60] et de meilleures propriétés différentielles que les fonctions Min et Max [64, 58]. Ces contraintes sont imposées par les domaines d'applications envisagés (modélisation d'objets hétérogènes, reconstruction d'objets et optimisation de formes), où la differentiabilité et l'approximation de la distance sont requis. Le but est de permettre la modélisation constructive d'objets définis par une fonction décrivant le champ de distances à la surface du solide.

- l'introduction d'un framework pour la modélisation constructive - un sous-ensemble du modèle FRep - utilisant les précédentes opérations ensemblistes et des primitives définies par la distance signée (ou son approximation).

- l'extension du modèle constructif hypervolumique (constructive hyper-volume model [51]) avec le précédent framework pour permettre la paramétrisation et le contrôle des attributs internes d'un solide par la distance Euclidienne. L'utilisation du framework précédent a pour but de corriger le principal défaut du modèle constructif hypervolumique, où les modèles construits à partir des R-fonctions, sont difficilement paramétrables pour définir les propriétés internes de l'objet.

- l'automatisation de la modélisation d'objets par l'introduction de modèles constructifs paramétriques et d'algorithmes pour optimiser ces modèles vers des objets définis par des nuages de points afin de faciliter la modélisation d'objets. Le but est de réutiliser des modèles templates, définis pour une famille d'objets, à différentes instances. Ces modèles templates sont définis de façon constructive et doivent être optimisés pour définir différentes instances d'objets.

- la proposition de nouveaux algorithmes, utilisant des algorithmes génétiques et

la programmation génétique, pour la création automatique de modèles constructifs (i.e. modèles construits comme une combinaison de primitives et d'opérations appliquées à ces primitives) à partir de nuages de points. L'automatisation de la construction d'un modèle constructif doit faciliter la création de modèles templates introduit dans ce manuscrit.

Les algorithmes et méthodes décrit dans ce manuscrit sont illustrés par des exemples issus de la modélisation d'objets mécaniques et la modélisation d'objets d'héritage culturel.

Ce manuscrit est présenté sous la forme d'une thèse article. Le chapitre suivant présente l'état de l'art du domaine de la construction et reconstruction volumique d'objets. Le troisième chapitre présente les contributions principales des recherches effectuées et relie chaque contribution aux publications jointes en annexe. Les publications, jointes en annexe, détaillent les démarches et résultats obtenus. Finalement, la conclusion récapitule les points importants de ces travaux et les perspectives envisagées.

CHAPITRE 2

État de l'art

Dans ce chapitre, nous rappelons brièvement les différents modèles utilisés pour représenter des objets solides ou volumétriques. En particulier, nous portons notre attention sur le modèle de la représentation fonctionnelle, abbrégé en FRep (l'acronyme de Function Representation) [52]. Comme nous allons le voir par la suite, ce modèle présente de nombreux avantages par rapport aux autres types de représentations.

Étant donné un solide S, la fonction qui à un point $\mathbf{p}$ associe la distance Euclidienne de $\mathbf{p}$ à la surface de S, est un exemple de FRep et apparaît comme une méthode concise et puissante pour la définition de solides : nous présentons par la suite différentes méthodes pour construire une fonction représentant un champ de distances à un solide ou un objet volumétrique.

La modélisation constructive est une méthode élégante pour le design et la création de solides à partir de simples primitives et d'opérations agissant sur ces primitives. La modélisation constructive peut être implémentée dans le modèle FRep en utilisant la théorie des R-fonctions [60, 62, 67, 52] où les fonctions min/max [67, 58] (nous remarquons que min/max sont un cas particulier de R-fonctions). Nous proposons un bref rappel dans la suite des méthodes et algorithmes pour créer des solides ou objets volumétriques avec la modélisation constructive, en soulignant la qualité d'approximation de la distance Euclidienne et la différentiabilité de la fonction définissant l'objet final.

La modélisation constructive d'objets définis par une approximation de la fonction distance a de nombreuses applications en modélisation de formes et de solides. La construction de solides avec une distribution hétérogène de matériaux en est un bon exemple : nous présentons par la suite différentes méthodes existantes pour la modélisation de solides ayant des distributions de matériaux hétérogènes.

La modélisation et construction de solides est une tâche complexe et répétitive et nous nous intéressons aux méthodes pour automatiser ce processus notamment par l'utilisation de modèles paramétriques. Nous allons tout d'abord présenter les méthodes existantes pour l'automatisation de la modélisation de formes et solides numérisés par

des scanners 3D. Le problème d'automatisation de la modélisation de solides inclue de nombreux problèmes connexes tels que : la segmentation de nuages de points, l'optimisation de primitives ou encore la construction du modèle final.

2.1 Modèles pour la représentation de solides

Tout d'abord, nous décrivons brièvement les différents modèles utilisés pour la représentation de solides par ordinateurs. Nous utilisons la même classification que celle utilisée par [70], qui propose principalement deux catégories :

- Représentation énumérative (et combinatoire),
- Représentation implicite (et constructive).

2.1.1 Représentation énumérative

2.1.1.1 Mapping

Les approches énumératives spécifient les règles nécessaires pour la génération de points appartenant au solide. L'approche la plus populaire dans cette catégorie est probablement l'utilisation de définitions paramétriques. Les points de l'espace paramétrique sont mappés vers l'espace géométrique par des fonctions. Les exemples classiques de représentation paramétriques sont les B-Splines, les patches de Bézier ou les B-Splines rationelles non uniformes (NURBS) [26]. Cette représentation est populaire en conception assistée par ordinateurs (CAO) mais a de nombreuses limitations :

- tester l'appartenance d'un point à un objet nécessite de complexes procédures numériques
- il est difficile de définir des opérations sur ces objets, comme par exemple les opérations ensemblistes

2.1.1.2 Groupement et énumération spatiale

Le groupement est la plus simple des méthodes pour représenter un ensemble : l'énumération consiste en une collection de cellules ayant même type géométrique et dimension. L'exemple le plus connu appartenant à cette catégorie est le voxel (une collection de cubes). Les octrees [14] et la partition spatiale binaire (Binary Spatial Partition ou BSP) [75] appartiennent aussi à cette catégorie. Le problème de cette représentation réside dans la difficulté de définir des opérations sur ces objets : les opérations ensemblistes ou même les translations ou rotations nécessitent des traitements particuliers.

2.1.1.3 Les complexes cellulaires

Les complexes cellulaires représentent une extension de la catégorie précédente. L'avantage de cette représentation est que l'information topologique du solide est intégrée à la représentation, de fait, aucun calcul supplémentaire n'est nécessaire pour répondre à des questions d'ordre topologique. Le premier inconvénient de ce modèle est le fait que l'information utilisée pour encoder le modèle est redondante. Le second est que la validité de la représentation doit être vérifiée à tout instant.

2.1.2 Représentation implicite

Au lieu d'énumérer tous les points appartenant à l'ensemble que nous voulons décrire, une autre méthode consiste à définir implicitement un ensemble géométrique S comme l'ensemble de points vérifiant un prédicat $P : S = \{\mathbf{p} : P(\mathbf{p}) == true\}$. L'un des prédicats les plus populaires consiste à utiliser le signe d'une fonction réelle f : par exemple l'inégalité $f(\mathbf{p}) \geq 0$ [60].

2.1.2.1 Géométrie constructive et géométrie du solide constructive

Le terme de géométrie constructive fait référence à la construction de solides en appliquant des opérations ensemblistes à des simples primitives. Cette méthode est implémentée en utilisant une structure de données de type arbre avec les opérations ensemblistes dans les noeuds de cet arbre et les primitives dans les feuilles. L'appartenance d'un point au solide se fait par un parcours récursif de l'arbre.

La géométrie de solide constructive ou CSG [57] est similaire au modèle décrit ci-dessus mais utilise uniquement des primitives régulières, et des opérations ensemblistes régularisées. Une primitive définie par un ensemble de points S est dite régulière si : $S = Fermeture(Intérieur(S))$. La régularisation des opérations ensemblistes nécessite de maintenir une information de voisinage pour les points sur les primitives et leurs combinaisons. Cela peut être difficile en pratique. Des systèmes de CAO complets basés sur cette théorie ont été implémentés [12] ; la simplicité des types de données et des algorithmes utilisés en ont fait une représentation populaire dans le milieu académique et dans certains systèmes commerciaux.

D'un point de vue pratique, ces méthodes permettent de garder un historique de la construction et de la modélisation, permettant entre autre d'éditer les éléments ou encore de refléter la structure logique de l'objet.

Un exemple d'arbre constructif est illustré par la figure 2.1. L'objet final est construit à partir de cylindres et de boîtes, illustrés dans les feuilles de l'arbre ; les différentes étapes de la construction de l'objet sont illustrés dans les noeuds internes

de l'arbre.

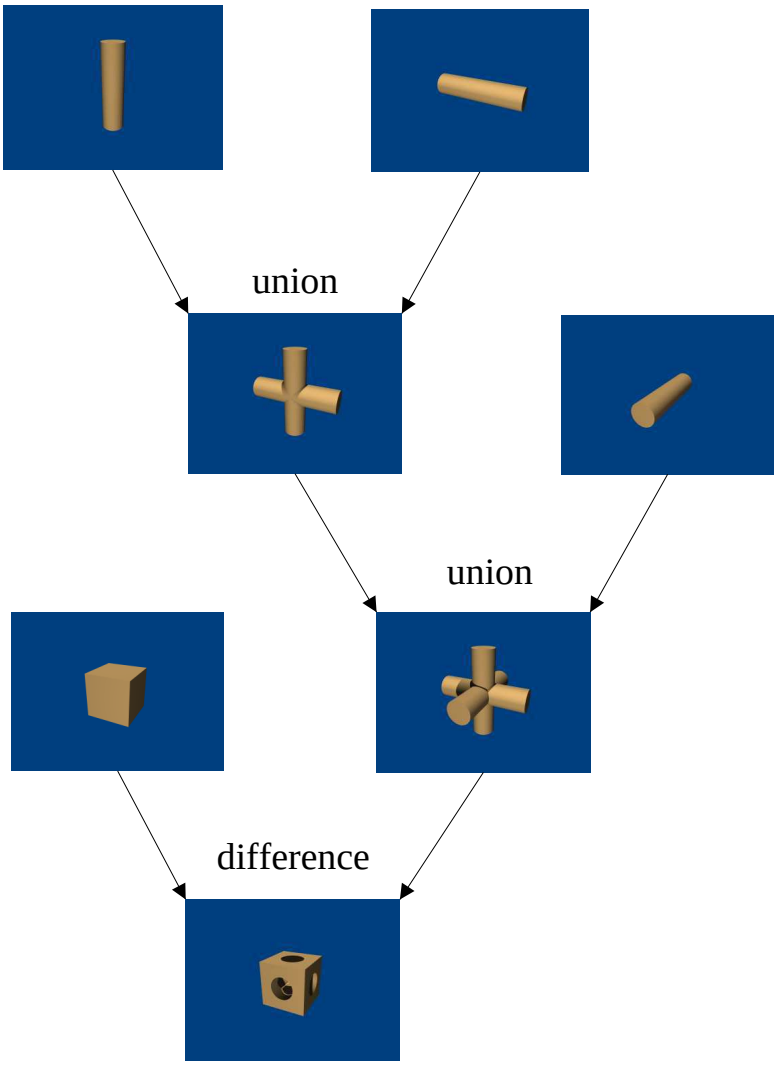


Figure 2.1 — Exemple d'arbre constructif avec les primitives dans les feuilles et les opérations dans les noeuds de l'arbre.

Un arbre constructif peut être traduit d'un point de vue syntaxique en une fonction réelle en utilisant la théorie des R-functions de Rvachev [60, 62, 67, 68, 52]. Cette théorie a été étendue par Pasko et al. pour former la théorie de la représentation fonctionnelle ou FRep (pour Function Representation) [52].

2.1.3 Représentation fonctionnelle

La représentation fonctionnelle ou FRep est une généralisation des surfaces implicites, de la géométrie constructive, et d'autres modèles pour la représentation de

solides ou de formes [52]. Dans le modèle FRep, un objet géométrique M est représenté par le signe d'une fonction. L'intérieur de M corresponds à $iM = \{\mathbf{p} \in E^n : f(\mathbf{p}) > 0\}$; la frontière de M corresponds à $\partial M = \{\mathbf{p} \in E^n : f(\mathbf{p}) = 0\}$; et l'extérieur de M à $eM = \{\mathbf{p} \in E^n : f(\mathbf{p}) < 0\}$, où E^n est l'espace Euclidien de dimension n.

Les objets de dimension deux et trois mais aussi des objets dépendant du temps ou encore des objets de dimensions supérieures peuvent être décrits par ce modèle [51]. En général, un objet est représenté par une structure en arbre (similaire à ce qui a été décrit pour la géométrie constructive) qui reflète la structure logique et la construction de l'objet. Les feuilles de l'arbre contiennent les primitives et les noeuds les opérations. La liste de primitives utilisables compte par exemple : les surfaces algébriques [11], les surfaces de convolutions [44], et autres. De nombreuses opérations ont été formulées qui gardent la représentation fermée, i.e. appliquer ces opérations sur un objet FRep crée un nouvel objet FRep. Quelques exemples d'opérations sont : les opérations ensemblistes [60, 64, 58], blending [54], offsetting [54], ...

La fonction de distance Euclidienne signée appartient au modèle FRep : si d est la distance Euclidienne signée à la surface du solide, alors l'intérieur du solide est défini par $\{\mathbf{p} : d(\mathbf{p}) > 0\}$, la frontière par $\{\mathbf{p} : d(\mathbf{p}) = 0\}$ et l'extérieur par $\{\mathbf{p} : d(\mathbf{p}) < 0\}$. Le modèle FRep est plus général et mélange la distance algébrique, Euclidienne et d'autres types de distances. Une comparaison entre les distances Euclidienne et algébriques à des surfaces quadriques suggère que la distance Euclidienne est préférable en terme de robustesse et de précision [29].

2.1.4 Représentation par frontières (BREP)

La représentation par frontières ou BRep est un compromis entre les différentes approches décrites précédemment. Un solide est défini implicitement par sa surface suivant le prédicat : $S = \{\mathbf{p} : \mathbf{p} \in$ l'ensemble fermé par $\partial S\}$ et repose sur le fait que ∂S sépare l'espace Euclidien E^3 en deux sous-ensembles : l'intérieur fermé par ∂S et l'extérieur non fermé. La requête utilisée pour déterminer l'appartenance d'un point à l'ensemble est obtenu en comptant le nombre d'intersections d'un rayon avec la surface du solide. La surface peut être représentée par n'importe quelle représentation qui garantit une définition non ambiguë et la possibilité de calculer l'intersection avec un rayon quelconque. La frontière du solide doit respecter les propriétés suivantes :

- être un complexe cellulaire valide,
- être un objet de dimension deux,
- chaque arête est partagé par un nombre pair de faces,
- être orientable (les normales sont orientées de manière globale et consistante).

Une manière naturelle d'implémenter le modèle BRep est d'utiliser une représentation combinatoire avec l'énumération des sommets, des arêtes et des faces ainsi que leurs relations (la topologie). Le modèle BRep est plus compact que le modèle cellulaire mais présente les mêmes défauts : difficile à construire et difficile à maintenir après des opérations.

2.2 Construction et approximation de la fonction de distance Euclidienne signée

2.2.1 La fonction de distance Euclidienne signée

La fonction de distance Euclidienne signée[1] d'un point $\mathbf{p} \in \mathbb{R}^n$ à un espace topologique de dimension $(n-1)$ fermé orientable M dans $\mathbb{R}^n$ est défini par : $d : \mathbb{R}^n \to \mathbb{R}$, $d(\mathbf{p}) = \epsilon|\mathbf{p} - \mathbf{c}|$, où ϵ vaut ± 1 correspondant à l'orientation de M, $\mathbf{c}$ est le point le plus proche sur M de $\mathbf{p}$, et $|.|$ est la distance Euclidienne. Deux conventions existent pour le signe de la distance : la normale extérieure peut être dirigée dans la direction positive ou dans la direction négative. Dans le cas d'un espace topologique de dimension $(n-1)$ non orientable ou non fermé, la distance signée n'a aucune véritable signification, mais la distance non signée peut être définie par : $d : \mathbb{R}^n \to \mathbb{R}$, $d(\mathbf{p}) = |\mathbf{p} - \mathbf{c}|$, en adoptant la même notation que précédemment. Dans ce manuscrit, nous adoptons la convention suivante : les normales pointent dans la direction des valeurs négatives de la fonction de distance. Dans l'espace Euclidien $\mathbb{R}^3$ de dimension 3, la fonction de distance Euclidienne signée à une surface orientée fermée M définit naturellement un solide par : $\{(x, y, z) \in \mathbb{R}^3 : d((x, y, z), M) \geq 0\}$.

La distance Euclidienne a de nombreuses applications : en modélisation géométrique [28], pour la métamorphose de formes [13], pour la reconstruction d'objets à partir des sections transversales [36], pour le rendu d'objets avec le tracé de sphère (sphere tracing) [30], pour la génération du squelette d'objets volumiques [84] entre autres.

2.2.2 Calcul de la distance

Soit $d(\mathbf{p})$, $\mathbf{p} \in \mathbb{R}^3$, la fonction de distance signée vers une surface fermée orientée M. La fonction d est la solution de viscosité de l'équation Eikonal [83, 77, 66] :

$$|\nabla d| = 1, d|_M = 0 \tag{2.2.1}$$

Physiquement, d correspond au temps d'arrivée d'une onde se propageant vers la surface, avec une vitesse de norme unitaire. Soit $\mathbf{c}$ le point le plus proche de $\mathbf{p}$ sur

[1] dans le reste de ce manuscrit, nous utiliserons le terme de fonction de distance pour se référer à la fonction de distance Euclidienne signée

la surface M, la distance est alors $|\mathbf{p} - \mathbf{c}|$, avec un signe négatif si $\mathbf{p}$ est en dehors de M. Si la surface est lisse, alors $\mathbf{p} - \mathbf{c}$ est orthogonal à la surface. La fonction de distance Euclidienne est au moins C^0, mais peut ne pas être dérivable en certains points.

Les expressions pour la distance à la plupart des surfaces classiques d'un système CSG (sphère, cylindre, cône) sont connues analytiquement [30]. Par exemple, la distance signée à une sphère (frontière d'une balle) de rayon 1 et centrée au point $(0, 0, 0)$ est donnée par la fonction : $d(x, y, z) = 1 - \sqrt{x^2 + y^2 + z^2}$. La distance signée à une ellipsoide peut être calculée par une procédure numérique [31]. L'algorithme proposé dans [31] n'est pas toutefois numériquement robuste. Supposons que l'ellipsoide soit centrée en $(0, 0, 0)$ avec e_x, e_y et e_z comme axes principaux, l'algorithme est numériquement instable pour tous les points proche ou sur les plans : $x = 0$, $y = 0$, ou $z = 0$. Cet algorithme a besoin d'être étendu pour traiter de façon robuste tous ces cas particuliers.

En général, si la surface M est défini comme un ensemble de points orientés (un couple : point, normal) ou un maillage de triangles, il est possible de résoudre l'équation Eikonal 2.2.1 sur une grille. Il existe plusieurs méthodes numériques pour résoudre ce problème telles que la méthode fast marching [66], la méthode fast sweeping [77, 83] ou l'algorithme de scan conversion [43]. Des algorithmes exploitant le processeur de la carte graphique (Graphics Processing Unit) ont été développés afin de calculer de façon efficace la fonction de distance Euclidienne [32, 74]. Une fois que la distance Euclidienne signée a été calculée en chaque point de la grille, il est possible d'utiliser des méthodes d'interpolation ou d'approximation par des splines afin d'obtenir une expression analytique – comme par exemple dans le travail de Roessl sur l'interpolation / approximation de données volumiques [59].

Ces méthodes peuvent avoir des problèmes de convergence et afficher de mauvaises performances en raison de certains facteurs comme le choix de la base, l'échantillonnage du champ discret de distance, ou la qualité des données en entrée (présence de bruit).

Les méthodes, décrites ci-dessus pour calculer le champ de distance définissant un objet, sont intéressantes pour des objets déjà existants, comme par exemple, des objets acquis par des scanners (scanner laser, IRM, CT). Ces méthodes associées à une interpolation nécessitent que l'objet, pour lequel la distance à la surface doit être calculée, existe déjà en tant que maillage de triangles ou en tant qu'ensemble de points orientés. Ce n'est pas la solution la plus simple lorsqu'il s'agit de modéliser un objet. En effet, cette méthode obligerait de produire une approximation par maillage de triangles de l'objet qui est en train d'être modéliser, puis de calculer la fonction de distance. Chaque modification au modèle, exigerait de re-calculer un maillage de la surface du solide puis de re-calculer le champ de distance associé. Quand un solide est modélisé par une approche constructive, il semble logique d'essayer d'établir la fonction de distance d'une manière constructive aussi, de la même manière qu'avec les R-fonctions.

La modélisation constructive de solides par des fonctions de distance, nécessite que les primitives soient définies par la fonction de distance, et que les opérations utilisées gardent la propriété de distance pour la fonction résultante. Les définitions existantes pour les opérations ensemblistes (union, intersection, différence) utilisées en modélisation constructive sont rappelés dans la section suivante.

2.2.3 Géométrie constructive avec des fonctions de distance

En géométrie constructive, des solides complexes sont construits par l'application successive d'opérations ensemblistes sur des primitives. Lorsque'un solide est décrit par le signe d'une fonction, comme par exemple pour les surfaces implicites ou FRep, les opérations ensemblistes peuvent être facilement définies. Quand les primitives sont définies par des champs de distance – i.e. la valeur de la fonction définissant le solide est la distance Euclidienne signée à la frontière du solide – nous voulons que la fonction résultante pour le solide, obtenue en appliquant les opérations ensemblistes à des primitives, soit encore la distance à la surface du solide construit (ou au moins une bonne approximation).

Nous étudions par la suite les différentes implémentations des opérations ensemblistes : min/max et les R-fonctions R_0 en termes d'approximation de la distance et de différentiabilité de ces fonctions.

Dans la suite de ce manuscrit, d_1 et d_2 sont deux fonctions de distance à deux ensembles de dimension $n-1$, M_1 et M_2 ; en pratique, $n = 2$ ou $n = 3$, et donc M_1 et M_2 sont des courbes ou des surfaces, et d_1 et d_2 définissent naturellement des surfaces ou solides, dénotés par S_1 et S_2. Les résultats restent valides pour toute dimension n.

2.2.3.1 min/max

Sabin [64] et Ricci [58], proposent indépendamment l'utilisation de min/max pour simuler des opérations ensemblistes sur des surfaces implicites. En utilisant min/max, les opérations ensemblistes (union, intersection, différence) sont définies par :

$$S_1 \cup S_2 = d_1 \vee d_2 = max(d_1, d_2) \tag{2.2.2}$$

$$S_1 \cap S_2 = d_1 \wedge d_2 = min(d_1, d_2) \tag{2.2.3}$$

$$S_1 \setminus S_2 = d_1 \wedge -d_2 \tag{2.2.4}$$

Il est facile de vérifier que la fonction construite en utilisant le min ou le max appliqué à deux fonctions de distance d_1 et d_2 vaut 0 sur la surface définie par

l'opération ensembliste appliquée sur les solides S_1 et S_2. Si le gradient de la fonction résultante est défini, sa norme est égale à 1 ; les deux propriétés de l'équation 2.2.1 sont vérifiées. Cependant, cette fonction n'est pas exactement la distance Euclidienne à la surface de l'objet construit. L'exemple 2D de la Figure 2.2 représentant la distance à un carré construit comme l'intersection de 4 lignes infinies illustre le problème. Les lignes de contour de la fonction de distance construite analytiquement par l'application du min sur les lignes infinies et la distance correcte à la frontière de l'objet sont illustrées à gauche et à droite respectivement. Les lignes de contour extérieures sont présentées en vert, les lignes de contour intérieures en bleu et l'objet construit en noir. La fonction de distance correcte a différentes lignes de contour extérieures en vert, avec des arcs circulaires centrés aux sommets de l'objet, au lieu de corners.

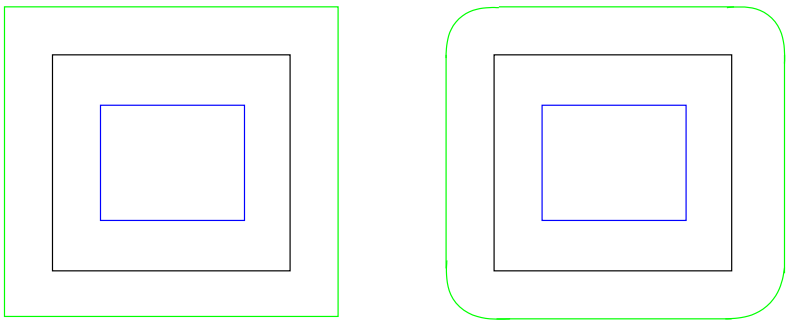


Figure 2.2 — Lignes de contour de : la distance à un carré unitaire, défini par l'intersection, en utilisant le min, de quatre lignes infinies (gauche), et la distance signée exacte à la frontière du carré (droite). Remarquons l'arc circulaire dans les lignes de contour extérieures (vert) de la distance exacte (droite), comparé aux corners créés en utilisant min (gauche).

Le principal problème avec l'utilisation de min/max dans la modélisation de formes et de solides est dû à la différentiabilité des fonctions min et max : $(x, y) \rightarrow min(x, y)$ et $(x, y) \rightarrow max(x, y)$ sont C^0 mais non différentiables aux points où $x = y$; par exemple, dans le cas de min : $min(x, y) = \frac{1}{2}(x + y - |x - y|)$, et $x \rightarrow |x|$ n'est pas différentiable en $x = 0$. Dans l'espace géométrique, la fonction résultante ne sera généralement pas différentiable pour les points p tels que : d_1 n'est pas différentiable, ou d_2 n'est pas différentiable, ou $d_1(p) = d_2(p)$. Les deux premiers cas sont inhérents aux primitives, mais le dernier est dû aux fonctions min / max.

Ces points peuvent causer des résultats inattendus en modélisation, par exemple lorsque certaines opérations sont appliquées à l'objet telle qu'une opération de blending, une opération définissant la métamorphose entre deux objets, et des problèmes pour des applications nécessitant l'existence (et la continuité) des gradients [7, 8].

La figure 2.3 illustre le résultat de l'union blending entre une sphère et un cube, quand le min/max est utilisé pour implémenter les opérations ensemblistes.

Le cube est défini comme l'intersection de plans. L'union blending est définie par : $blending(d_1, d_2) = d_1 + d_2 + \sqrt{d_1^2 + d_2^2} + \frac{a_0}{1+(\frac{d_1}{a_1})^2+(\frac{d_2}{a_2})^2}$ comme défini dans le travail de Pasko[54]. L'arête non désirée apparaissant dans l'objet vient de l'utilisation du min pour implémenter l'intersection lors de la définition du cube.

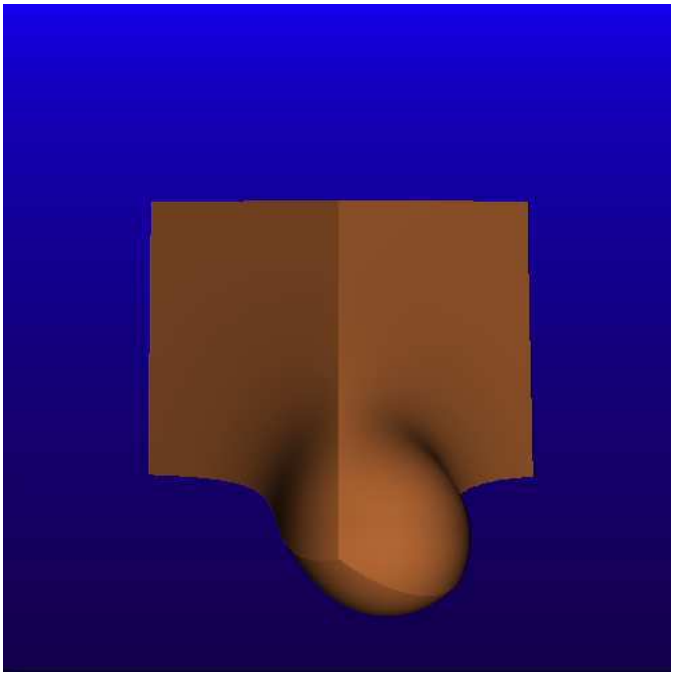


Figure 2.3 — Illustration de la discontinuité C^1 du min/max lors de la modélisation d'objets, dans cet exemple : l'union blending entre une sphère et un cube.

Ricci [58] a proposé des approximations super-elliptiques de min/max afin d'enlever les discontinuités C^1. Ces fonctions ne décrivent pas exactement les opérations ensemblistes et sont surtout utilisées pour des opérations de blending. L'approximation elliptique du min/max par Barthe [2] est initialement conçue pour des opérations de blending et l'erreur de l'approximation de la distance s'accroît loin de la surface. Rvachev a proposé les R-fonctions [60, 61, 67], qui sont discutées par la suite.

2.2.3.2 R-functions

Il existe différents types de R-fonctions, avec différents ordres de différentiabilité, discutées dans les papiers suivants [60, 62, 67]. La version la plus utilisée est définie par :

$$S_1 \cup S_2 = d_1 \vee d_2 = d_1 + d_2 + \sqrt{d_1^2 + d_2^2} \tag{2.2.5}$$

$$S_1 \cap S_2 = d_1 \wedge d_2 = d_1 + d_2 - \sqrt{d_1^2 + d_2^2} \tag{2.2.6}$$

$$S_1 \setminus S_2 = d_1 \wedge -d_2 \tag{2.2.7}$$

Les R-fonctions, $(x, y) \rightarrow x \wedge y$ et $(x, y) \rightarrow x \vee y$, sont C^1 sur $\mathbb{R}^2 \setminus (0.0)$. Dans l'espace géométrique – i.e. quand les R-fonctions sont appliquées aux primitives d_1 et d_2 – la fonction résultante n'est pas différentiable en tous points p tels que : d_1 n'est pas différentiable, ou d_2 n'est pas différentiable, ou $d_1(p) = d_2(p) = 0$. Les deux premiers cas sont inhérents aux primitives et le dernier est dû aux R-fonctions. Ces points correspondent aux corners et arêtes d'une surface. Lorsque les R-fonctions sont utilisées pour implémenter les opérations ensemblistes, l'opération d'union avec blending ne crée pas d'arête non désirée comme dans la figure 2.3 et produit une forme lisse comme dans la figure 2.4.

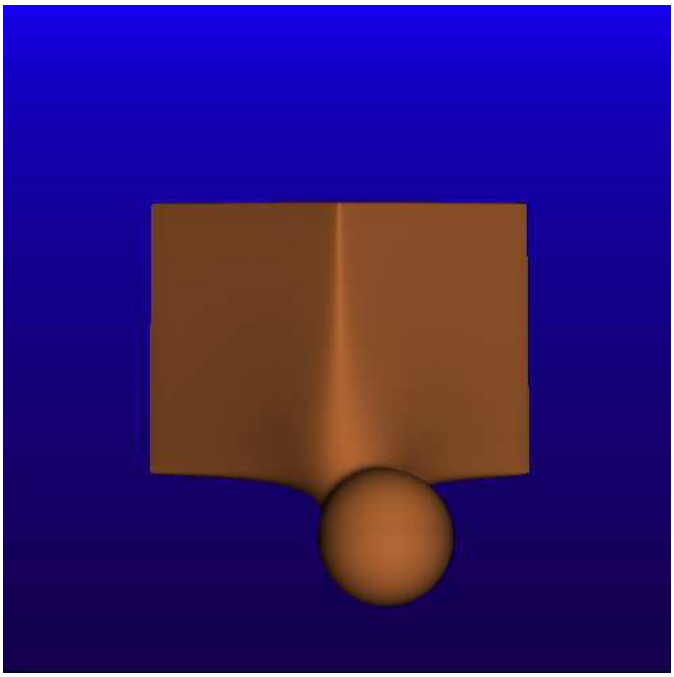


Figure 2.4 — Illustration d'un effet de blending, quand des R-fonctions sont employées en tant qu'opérations ensemblistes durant la modélisation.

Si l'ordre C^k est nécessaire, les R-fonctions suivantes peuvent être utilisées :

$$S_1 \cup S_2 = d_1 \vee d_2 = d_1 + d_2 + (d_1^k + d_2^k)^{\frac{1}{k}} \tag{2.2.8}$$

$$S_1 \cap S_2 = d_1 \wedge d_2 = d_1 + d_2 - (d_1^k + d_2^k)^{\frac{1}{k}} \tag{2.2.9}$$

$$S_1 \setminus S_2 = d_1 \wedge -d_2 \tag{2.2.10}$$

Les R-fonctions sont cependant une mauvaise approximation de la distance. Elles souffrent de l'explosion des valeurs lorsque par exemple elles sont appliquées à des solides s'intersectant ; la figure 2.5 illustre la différence entre la distance à un cercle de rayon 5 résultant de l'union du disque avec lui-même quand l'union est définie par une R-fonction et la distance au cercle : l'union atteint une valeur de 17.071 au centre contre 5 pour le disque seul.

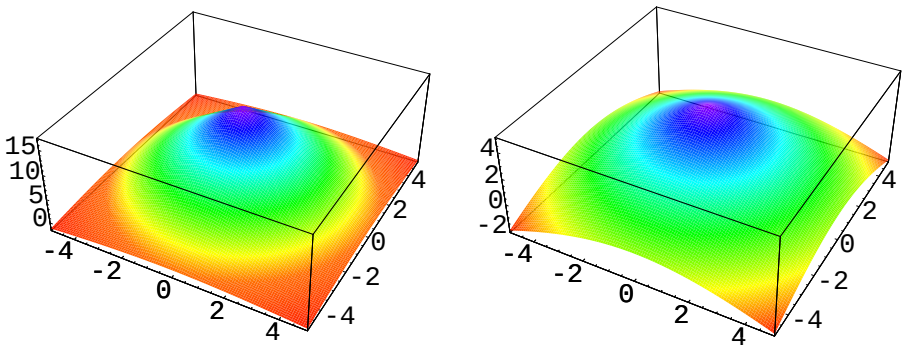


Figure 2.5 — Gauche, « distance » à un cercle de rayon 5 obtenu par l'union du disque avec lui-même, le maximum de la fonction atteint 17.071 ; Droite, distance à un cercle du rayon 5, la valeur maximum est 5, atteinte au centre.

2.2.3.3 Discussion

Ni le min/max, ni les R-fonctions ne fournissent en même temps une approximation raisonnable de la distance et une continuité des dérivées de la fonction résultante. Les R-fonctions sont une mauvaise approximation de la distance Euclidienne, et min/max introduit des discontinuités dans le gradient de la fonction (voir la figure 2.2). Notons cependant que différentiabilité et exactitude de la distance sont contradictoires, puisque la fonction de distance est par définition non-différentiable en certains points (elle n'est pas différentiable en tous points appartenant à l'axe médian du solide). Nous acceptons de perdre en exactitude en faveur de la différentiabilité pour certaines applications.

Nous présentons dans ce manuscrit de nouvelles formulations pour les opérations ensemblistes inspirées par les travaux suivants [2, 34]. Les fonctions proposées sont C^1 et proposent une approximation contrôlable de la distance Euclidienne. De ce point de vue, nous appelons la fonction construite par le terme de fonction réelle de distance approximée signée (SARDF acronyme de Signed Approximate Real Distance Function), la fonction correspondant à l'intersection est appelée intersection SARDF,

et la fonction correspondant à l'union appelée union SARDF.

Ces opérations ensemblistes forment la base du système de modélisation avec les primitives définies par la fonction de distance signée ou une approximation. Des opérations de blending peuvent être dérivées à partir des opérations ensemblistes, et d'autres opérations isométriques (rotation, translation, etc) les complètent.

2.3 Modélisation de matériaux hétérogènes

2.3.1 Techniques pour la modélisation de matériaux hétérogènes

Les méthodes de modélisation de solides sont la plupart du temps restreintes à la définition de modèles qui capturent seulement la géométrie des objets, sous l'hypothèse que ces objets sont homogènes. Récemment, une attention particulière a été prêtée à la modélisation d'objets hétérogènes. Lors de la modélisation d'objets hétérogènes, un objet a un certain nombre d'attributs non-uniformément distribués et assignés à chaque point et variant dans l'espace. Ces attributs peuvent être ou ne pas être continus et sont de natures différentes telles que photométrique, ou représentant la densité ou la distribution de matériau, etc. La modélisation d'objets hétérogènes a de nombreuses applications en CAD/CAM/CAE, en prototypage rapide, pour des simulations physiques, en modélisation géologique ou médicale.

Plusieurs techniques pour modéliser des objets hétérogènes ont été développés, présentant certaines analogies apparentes avec les représentations utilisées pour la modélisation d'objets homogènes présentées dans la section 2.1.

Les R-set sont considérés à la base pour la modélisation et étendus pour l'inclusion de matériau dans [39]. Un objet est subdivisé en composantes ; chacune est homogène à l'intérieur et on lui affecte un index de matériau. Des opérations ensemblistes peuvent être appliquées aux composantes du solide avec les opérations correspondantes sur les matériaux. Malheureusement, de telles techniques de modélisation sont limitées à la représentation d'objets dont les propriétés (attributs) varient discrètement. Dans [38], un modèle plus général est proposé : la géométrie est représentée par la décomposition de l'ensemble de points par un ensemble fini de 3 cellules fermées, tandis que les attributs sont définis par une collection de fonctions, qui mappent la géométrie de l'objet vers plusieurs attributs. Un tel modèle mathématique est connu sous le nom de faisceau de fibres, avec le modèle géométrique jouant le rôle de l'espace de base. Plusieurs autres travaux emploient le même modèle, avec des extensions dans différentes directions ([6, 17]). Cependant, comme remarqué dans [8], un tel modèle n'est pas vraiment facile à implémenter.

Les représentations volumétriques définissent naturellement des solides. Un objet homogène peut être défini comme un sous-ensemble de l'espace à trois dimensions, avec une valeur scalaire additionnelle en chaque point. Dans le cas d'une énumération spatiale, comme les voxels [56], l'extension pour modéliser des objets hétérogènes consiste à ajouter une valeur scalaire pour chaque attribut [47]. L'inconvénient de cette méthode est la difficulté pour décrire la distribution des attributs, sans utiliser un dispositif d'acquisition de données (par conséquent, il est supposé que l'objet à modéliser existe déjà). En outre, le caractère discret du modèle exige certaines procédures spécifiques d'approximation.

Une représentation volumétrique continue a été proposée dans [55], où un volume de B-spline est utilisé pour modéliser la géométrie d'objets, tandis que les attributs sont modélisés par diffusion. Ce modèle souffre d'un manque de flexibilité puisque la géométrie des objets est limitée par l'utilisation de splines volumiques.

Le modèle hypervolumique constructif est présenté dans [51] comme un modèle mathématique pour définir des objets hétérogènes. Un objet hypervolumique est défini comme un ensemble de points multi-dimensionnels G avec des attributs multiples en chacun de ses points. Les attributs S_i représentent des valeurs abstraites ou des caractéristiques physiques telles que la température, la couleur, la distribution du matériau, etc. La représentation d'un hypervolume est définie par :

$$o = (G, A_1, \ldots, A_k) : (F(X), S_1(X), \ldots, S_k(X))$$

où :

- $X = (x_1, \ldots, x_n)$ est un point de l'espace Euclidien E^n de dimension n,
- $F : E^n \to \mathbb{R}$ est une fonction à valeurs réelles utilisée pour représenter l'ensemble de points G, et basée sur le modèle FRep [52]
- $S_i : SP_i \to \mathbb{R}$, $SP_i \subset E^n$ est une fonction scalaire à valeurs réelles correspondant à un attribut A_i, qui n'est pas nécessairement continue.

La fonction $F(X)$ est une fonction à valeurs réelles. Pour chaque point, la fonction est évaluée et selon le signe de la valeur retournée, on peut classifier le point comme étant à l'intérieur, à l'extérieur ou sur la surface de l'objet. Cette fonction est représentée par une structure arborescente avec des primitives dans les feuilles de l'arbre et des opérations dans les noeuds. La seule condition du modèle FRep est que la fonction définissant F doit être au moins C^0 continu.

De même, selon les applications, les fonctions utilisées pour décrire les attributs peuvent être définies en utilisant des modèles physiques ou une approche constructive. Le sous-ensemble spatial où un attribut est défini, s'appelle une *partition spatiale* indiqué par SP_i dans la formulation ci-dessus. Il n'y a aucune valeur définie pour

un attribut en dehors de sa partition de l'espace. Pour chaque attribut, il y a au moins une partition de l'espace contenant cet attribut matériel. Toutefois, un attribut matériel peut être contenu dans plus d'une partition de l'espace, dans le cas par exemple où un attribut est défini par la composition connue de plusieurs matériaux.

La géométrie et les partitions spatiales de l'objet peuvent être définies par modélisation constructive, en utilisant les R-fonctions générales [60, 61], ou les fonctions min / max [58, 64]. Cependant, le problème de la paramétrisation et du contrôle des attributs n'est pas présenté dans ce modèle.

Biswas et al. [8] s'intéressent à la représentation et le contrôle des distributions de matériau par des paramètres intuitifs liés à la géométrie du solide et/ou de ses attributs matériels pour la modélisation d'objets. Ils proposent d'employer des fonctions de la distance aux attributs du matériau (définis comme des ensembles de points de n'importe quelle dimension avec des propriétés connues) en tant que paramètres. Il apparaît en effet dans la littérature que la distance (Euclidienne), ou les fonctions de la distance sont les fonctions les plus utilisées pour décrire des distributions de matières par les méthodes basées sur la discrétisation spatiale [73, 35, 40]. Les auteurs de [8] montrent également que cette approche est théoriquement complète car elle permet de représenter toutes les distributions possibles de matière. Cependant, dans leur travail, la modélisation de la géométrie des solides et des attributs par des champs de distance Euclidienne n'est pas présentée.

2.3.2 Discussion

Nous proposons d'utiliser les formulations des opérations ensemblistes et le modèle SARDF dans le modèle hypervolumique constructif [51] discuté précédemment. Dans ce modèle, à la fois la géométrie du solide et la géométrie des domaines où sont définis les attributs vont être définis de façon constructive en utilisant les primitives et les opérations SARDF.

Le modèle hypervolumique constructif modifié est utilisé pour répondre à la question (section 5.2 de [8]) sur les méthodes pour construire un champ de distance Euclidienne et combiné au travaux [51, 8], il sert ensuite à modéliser des objets constructifs hétérogènes en utilisant des fonctions de distances signées.

2.4 Automatisation de la modélisation d'objets et rétro-conception (reverse engineering)

La modélisation de solides est une tâche répétitive et difficile. Nous examinons comment automatiser le processus de modélisation. Une automatisation est nécessaire par exemple lors de la reconstruction d'objets réels acquis par un scanner laser. Ce processus, appelé rétro-conception (reverse engineering), nécessite les étapes suivantes :

- acquisition des données ; la méthode la plus répandue consiste à utiliser un scanner laser, mais il existe de nombreuses autres méthodes [80]
- prétraitement ; comme la suppression du bruit, le calcul des normales orientées globalement et de façon cohérente, la combinaison de vues multiples obtenues à partir de différentes acquisitions de données
- la segmentation et l'ajustement de surface, où les points sont regroupés en sous-ensembles de points pour lesquels une surface adéquate est déterminée
- création d'un modèle géométrique

L'automatisation de la modélisation est aussi nécessaire pour convertir un modèle d'un format à un autre, par exemple d'un modèle BRep vers un modèle CSG.

Généralement, il est possible de distinguer la reconstruction de solides pour des applications de type informatique graphique et de type conception assistée par ordinateur (CAO). La conception assistée par ordinateur impose plus de contraintes sur le solide à reconstruire.

2.4.1 Reconstruction de maillages et de surfaces implicites

Reconstruction de maillages : Hoppe [33] reconstruit une surface en trois étapes : 1) estimation d'une surface initiale en calculant une fonction de distance signée, 2) simplification du maillage, 3) optimisation du maillage. La méthode Power Crust d'Amenta [1] repose sur une approximation de la transformation de l'axe médian (Medial axis transform) avec des sphères.

Ohtake et al. [50] calculent une approximation du maillage avec les étapes suivantes : 1) création d'une couverture adaptive du nuage de points, 2) création de points auxiliaires, 3) nettoyage du maillage, traitement des trous présents dans le maillage et optimisation du maillage.

Reconstruction de surfaces implicites : Les surfaces implicites, ou FRep, ont été utilisées pour générer des surfaces à partir d'un nuage de points. Une approximation du maillage de la surface peut ensuite être créée à partir de la surface implicite en utilisant des algorithmes de polygonisation comme le Marching

Cube [41], ou d'autres algorithmes pour la polygonisation de surfaces implicites [10, 53].

La méthode de reconstruction proposée par Muraki [46] consiste à optimiser des "blobby models" [9] au nuage de points. Savchenko et al. [65] puis Turk et al. [78] proposent une combinaison linéaire de fonctions de bases radiales optimisée pour reconstruire le nuage de points. Les fonctions de bases radiales compactes ont été introduites par Morse et al. [45] pour réduire la complexité en temps et en mémoire des méthodes précédentes. La méthode "Partition of Unity" (MPU pour Multi-level Partition of Unity) a été introduite par Ohtake et al. [48] comme un moyen élégant pour partitionner le nuage de points, reconstruire des surfaces implicites localement et unifier globalement les différentes surfaces implicites locales, permettant de réduire fortement la complexité en temps et en mémoire de l'algorithme de reconstruction.

Ces méthodes produisent des modèles extrément compliqués, même dans le cas de surfaces implicites qui contiennent un grand nombre de coefficients pour les splines ainsi que l'ensemble des points appartenant au nuage de points original. Ces méthodes sont difficiles à utiliser pour inspecter ou réutiliser la structure des objets contrairement à des méthodes de géométrie constructive. Par ailleurs, à part les travaux d'Amenta [1], ces travaux manquent de garanties sur la surface reconstruite, comme par exemple : l'existence de fissures et de trous dans le maillage reconstruit, une topologie différente de celle du modèle original.

Ces méthodes sont cependant très efficaces pour produire des maillages pour des modèles compliqués, pleins de détails, afin de les utiliser dans différentes applications telles que des jeux, des musées virtuels ...

2.4.2 Reverse engineering pour la conception assistée par ordinateur (CAO)

Les objectifs du reverse engineering pour le CAO sont similaires à ceux décrits précédemment : générer un modèle CAO pour des objets existants. Cependant, il impose des critères différents : le modèle final doit être un modèle CAO valide permettant la manipulation du solide et la possibilité d'effectuer des opérations ultérieures.

La principale différence entre le reverse engineering pour la CAO et pour des applications de type informatique graphique est que la création de modèles géométriques CAO impose des contraintes sur la précision et la cohérence des modèles reconstruits, qui doivent utiliser des surfaces issues de systèmes CAO [80]. Le modèle standard utilisé est le modèle BRep.

Les problèmes typicalement rencontrés sont l'identification d'arêtes vives, le

traitement des blend, garantir la continuité entre les différents patchs constituant le modèle. Une autre difficulté est le respect dans le modèle reconstruit de différentes contraintes issues du solide original, comme par exemple : maintenir le parallélisme entre des plans, la concentricité de sphères ... [4].

La segmentation, qui est habituellement omise dans les méthodes de reconstruction pour des applications en informatique graphique, est un problème particulièrement difficile. Il consiste à regrouper les points du nuage original en différents sous-ensembles correspondant à des primitives constituant le modèle. Il s'agit d'une étape importante pour identifier la structure logique du modèle CAO final. La thèse de Vanco [79] étudie le problème de la segmentation directe de l'ensemble de départ. Elle repose sur une estimation des normales et de la courbure du solide, la segmentation du nuage de points est ensuite effectuée par regroupement des points en utilisant les informations apportées par les normales et la courbure du solide en chaque point. Des primitives simples sont optimisées en utilisant ces information et aident à produire un résultat de segmentation plus robuste.

L'inconvénient majeur de cette approche est que le calcul précis des normales et de la courbure en chaque point est un problème très difficile en pratique. Le calcul de normales avec une orientation cohérente est NP complet. Hoppe [33] propose une heuristique pour résoudre ce problème. Le calcul fiable de la courbure du solide en chaque point est un problème encore plus difficile [49, 81, 16].

Benko et Varady proposent aussi une méthode basée sur une segmentation directe en proposant une série de simple tests pour segmenter l'ensemble de points original [5]. Marshall et al. [42] segmentent le nuage de points par des primitives élémentaires (sphere, tore, plan) définis par une approximation de la distance Euclidienne.

Chevalier et al proposent de segmenter le nuage de points à l'aide de superellipsoides. L'objet reconstruit est alors défini par l'union des superellipsoides calculées précédemment [18].

Après la segmentation de l'ensemble de points, des primitives sont associées à chaque sous-ensemble. Parfois, cette étape fait partie de la segmentation [42, 18].

L'étape finale est la création du modèle CAO. Cela consiste à regrouper les différentes primitives en un seul solide. Quelques unes des difficultés majeures proviennent des contraintes à satisfaire dans le modèle final, comme par exemple l'orthogonalité ou le parallélisme de plans, la continuité entre les différentes primitives du modèle .... Un autre problème, lorsque le modèle utilisé pour la représentation est un modèle BRep est la création d'un modèle valide, et nous avons vu dans la section 2.1 que cela peut être difficile à satisfaire.

2.4.3 Transformation BRep - CSG

Un problème important en modélisation de solides, mais qui existe aussi dans d'autres domaines de l'informatique, est la conversion de données d'une représentation vers une autre. Convertir des modèles BRep en CSG a de nombreuses applications pratiques mais est malheureusement un problème difficile à résoudre.

Le problème de la conversion de modèle BRep vers un modèle CSG a été étudié en premier par Rvachev et al. en deux dimensions [63], où un algorithme est proposé pour convertir une forme définie par un polygone linéaire en dimension deux en une représentation CSG. Dans la littérature, un algorithme similaire est attribué à Batchelor [3], différentes versions et améliorations de ces algorithmes ont été proposées et décrites dans [82, 76, 19]. Ces algorithmes sont basés sur le concept suivant : un polygone linéaire peut être représenté par la différence entre son enveloppe convexe et un nombre fini de concavités.

Shapiro a amélioré cet algorithme afin de pouvoir traiter des polygones ayant des courbes [69]. Des algorithmes similaires ont été adaptés pour le cas de polyèdres à trois dimensions, mais malheureusement ces algorithmes ne fonctionnent pas avec des polyèdres quelconques.

En dimension trois, le problème a été résolu pour des solides limités par des surfaces de second degré [71, 15]. L'algorithme nécessite l'addition de surfaces implicites qui n'apparaissent pas avec l'information surfacique.

2.4.4 Discussion

Les méthodes générant un maillage approximant la surface présentent des problèmes pour certaines applications utilisant le modèle reconstruit. Le maillage généré ne fournit aucune garantie théorique sur la précision ou la cohérence de l'objet reconstruit. Les mêmes problèmes existent pour les méthodes basées sur la reconstruction de surfaces implicites. Dans la plupart des cas, le modèle reconstruit ne présente aucune garantie de précision ou de cohérence, comme par exemple, l'absence d'isosurfaces supplémentaires, ou l'existence de surfaces non connectés avec le reste du solide.

Reconstruire des objets par des surfaces implicites semble préférable à la reconstruction de maillages puisque cette méthode fournit une fonction définissant l'objet qui peut ensuite être utilisée comme un modèle FRep valide. Cette fonction peut par exemple être utilisée pour générer un maillage adapté à différentes applications comme par exemple des méthodes de calcul par éléments finis [37] ou encore être utilisée directement pour des méthodes de calcul sans maillage [27].

La rétro-conception pour des applications de type CAO a des besoins plus contraignants que pour des applications de type infographie : elle a pour objectif de fournir des modèles avec certaines garanties de cohérence et de précision. Dans la majorité des applications, le modèle reconstruit est de type BRep, et présente certains problèmes comme ceux décrits dans la section 2.1. Pour certaines applications, il est intéressant de posséder un arbre de construction, car il peut être utilisé ensuite pour inspecter ou modifier la structure de l'objet reconstruit.

Le même besoin d'un arbre constructif existe pour les surfaces implicites. Lors de la reconstruction d'un modèle FRep, un objet défini par un arbre constructif contient plus d'information sémantique qu'une fonction reconstruite comme somme de splines et leurs coefficients.

Un arbre constructif donne la possibilité de traiter ultérieurement l'objet reconstruit. Il permet aussi d'inspecter la structure de l'objet, modifier les primitives ou les opérations pour modifier le modèle. La partie gauche de la figure 2.6, illustre l'importance d'un arbre constructif pour le modèle reconstruit. En supposant que nous voulions éditer le modèle et créer un nouveau modèle comme celui de la partie droite de la figure 2.6 ; si le modèle est disponible sous la forme d'un modèle constructif, il est facile de remplacer l'opération ensembliste par une opération de blending. Il serait plus difficile de réaliser la même opération si l'objet était défini par une représentation de type BRep ou par un ensemble de segments.

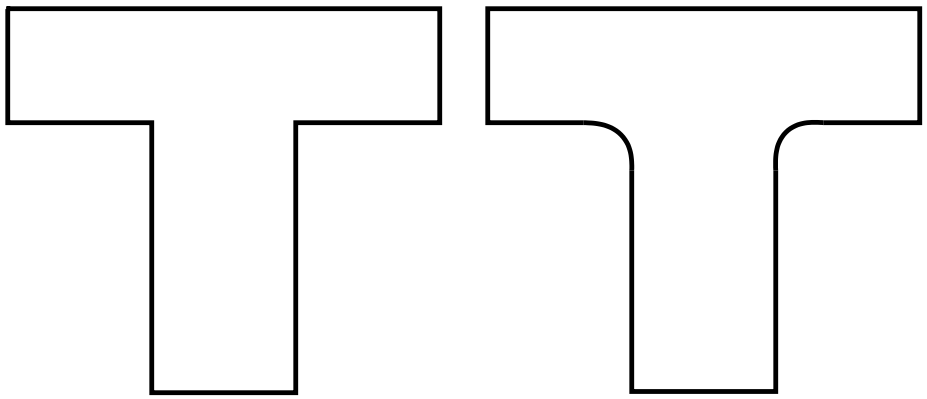


Figure 2.6 — A gauche, une simple forme en T constituée de l'union de deux blocs. A droite, l'opération union est remplacée par un blending.

La possibilité de transformer un arbre constructif en une fonction réelle est une seconde nécessité. Comme expliqué précédemment, une fonction réelle peut être utilisée pour des traitements analytiques ultérieurs comme par exemple un remaillage adapté à des méthodes de calcul aux éléments finis [37], ou des méthodes de calcul sans maillage [27].

En prenant en compte les critères énumérés ci-dessus, nous considérons le problème de la rétro-conception d'objets définis par des modèles FRep constructifs. Pour cela, nous introduisons l'idée de modèle FRep template. Un modèle FRep template est un modèle général pour une famille d'objets, où l'arbre constructif contient seulement des opérations et le type des primitives, tandis que les valeurs des paramètres utilisés pour définir les opérations et les primitives ne sont pas définis mais sont optimisés pour chaque objet. Ces paramètres gouvernent la forme du solide. La modification de ces paramètres peut résulter en différentes formes qui sont optimisées pour satisfaire certains critères de modélisation.

Des modèles templates peuvent exister dans des bibliothèques spécialisées pour chaque domaine d'application : conception mécanique, conception de prothèse médicale, ou bien peuvent être créé par l'utilisateur. Pour un domaine donné d'application, différents objets peuvent avoir une forme similaire qui peut être décrite par un modèle template.

La création de modèles templates reste cependant une tâche difficile. Nous examinons l'automatisation de cette tâche en générant directement un modèle constructif FRep à partir d'un nuage de points. Les paramètres gouvernant la forme de l'objet peuvent être définis ensuite par l'utilisateur ou à partir d'études statistiques pour générer un modèle FRep template pour l'objet considéré.

CHAPITRE 3

Contributions

Dans ce chapitre, nous résumons les principales contributions de nos travaux et référons vers les articles décrivant en détail les algorithmes proposés et leurs résultats.

Les principales contributions présentées dans ce document sont :

- La définition de nouvelles fonctions et leurs propriétés pour l'implémentation d'opérations ensemblistes (intersection, union, différence), qui sont au moins C^1, à l'exception de l'origine, et forment une bonne approximation de la fonction de distance Euclidienne [25, 23],
- L'introduction d'un nouveau framework utilisant les opérations ensemblistes précédentes avec des primitives définies par la distance signée Euclidienne [25],
- L'utilisation du framework précédent pour la modélisation d'objets hétérogènes [23],
- L'introduction du concept de FRep template et la présentation d'algorithmes pour l'optimisation de modèles FRep templates [21, 20, 24],
- L'introduction d'algorithmes pour la création de modèles FRep constructifs à partir d'un nuage de points [22, 72].

3.1 SARDF framework

3.1.1 Objectifs

Le premier objectif de la thèse est de proposer une extension du framework FRep dans lequel les objets modélisés sont définis par la fonction distance Euclidienne. Ce framework doit permettre la modélisation constructive d'objets comme dans le modèle FRep. Il apparaît donc normal de proposer une modification des opérations et primitives existantes dans le modèle FRep qui permet de conserver le champ de distance.

3.1.2 Les opérations SARDF

Pour cela, nous introduisons dans ce manuscrit des constructions pour de nouvelles formulations des opérations ensemblistes : intersection, union et différence appelée SARDF (un acronyme pour Signed Approximate Real Distance Function) [25, 23].

Ces fonctions sont construites à partir des lignes de champs des fonctions min/max qui sont modifiées à leur intersection par un arc de cercle ou une partie d'ellipse, pour enlever le point de discontinuité des dérivés partielles, apparaissant à chaque intersection des lignes de champ parallèles aux axes x et y (figure 3.1).

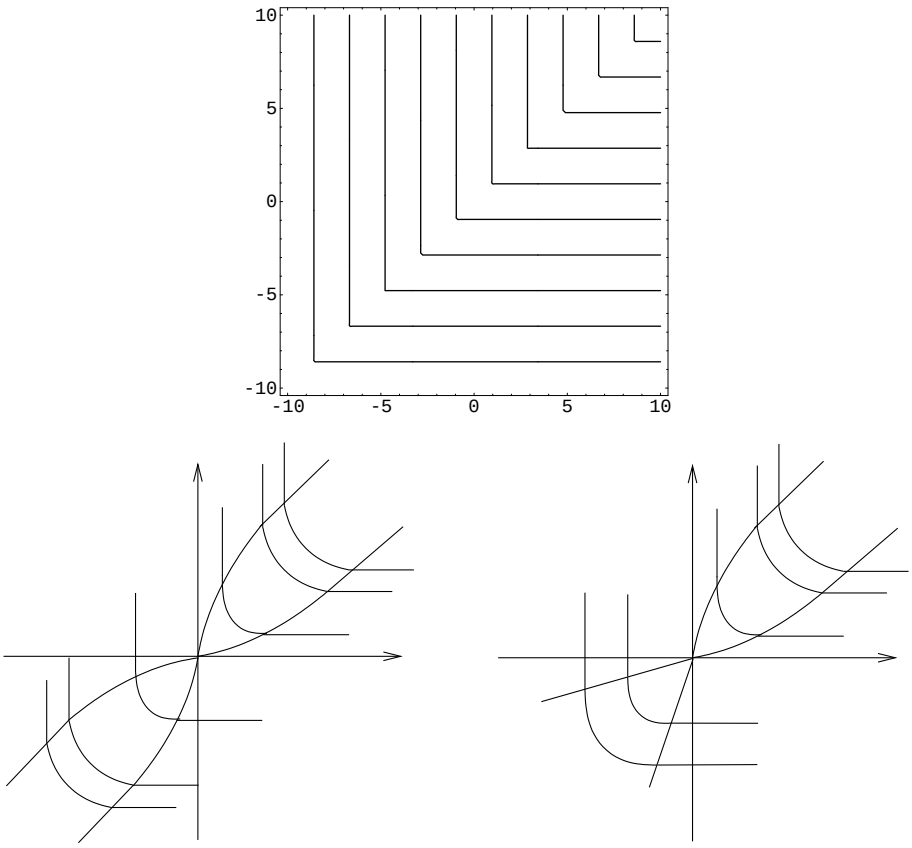


Figure 3.1 — Chaque ligne de champ de la fonction min possède un point de discontinuité (image du haut). Illustration de deux approches différentes pour la construction de l'intersection SARDF : limitation du rayon par deux rayons (bas, gauche) ; croissance illimitée du rayon (bas, droite).

Ces fonctions permettent une bonne approximation de la distance Euclidienne, lorsqu'elles sont utilisées avec des primitives définies par la distance Euclidienne. Par ailleurs, ces fonctions sont de classe C^1 à l'instar de min/max. En effet, l'introduction de discontinuité des dérivées partielles par les opérations ensemblistes posent des problèmes pour certaines opérations géométriques, du type blending, ainsi que des problèmes pour certaines applications [8].

3.1.3 Construction des opérations SARDF et implémentation du framework SARDF

La construction des opérations SARDF ainsi que leurs propriétés sont présentées en détails dans [25, 23]. Le framework SARDF est proposé et détaillé dans [25] : il est constitué des opérations SARDF, des opérations isométriques et de primitives définies par la distance Euclidienne ou une approximation. Une implémentation du framework SARDF dans une applet Java™est proposée et détaillée [25]. Dans cette application, les opérations et primitives SARDF sont implémentées à l'intérieur du language de modélisation de formes HyperFun. Les scripts HyperFun, avec l'extension permettant de définir des objets définis par la distance Euclidienne sont compilés en bytecode pour la machine virtuelle Java™.

Nous proposons aussi la modification des opérations SARDF pour définir certains types de blending [25]. Il est en effet très facile d'obtenir des opérations de blending avec addition ou soustraction de matériel par de simples modifications des opérations SARDF. L'idée consiste à utiliser un arc de cercle ou une partie d'ellipse pour chaque ligne de champs des opérations SARDF. Les opérations ensemblistes SARDF sont obligées de garder une discontinuité des dérivés partielles à l'origine pour pouvoir modéliser des arêtes.

3.1.4 Résultats

Un des objectifs du framework SARDF est la modélisation de solides par des champs de distances approximant la distance Euclidienne. En effet, la distance Euclidienne sert de paramètre pour certaines applications comme la modélisation de solides hétérogènes, la définition de transformation ou de métamorphoses entre objets .... Les R-fonctions utilisées généralement dans le modèle FRep proposent une mauvaise approximation de la distance Euclidienne, qui rend le modèle FRep difficile à utiliser pour certaines applications. Les opérations SARDF ont été construites pour proposer une meilleure approximation du champ de distance que les R-fonctions ou min/max.

Les opérations SARDF sont qualitativement comparées aux opérations min/max et aux R-fonctions, utilisées traditionellement en modélisation constructive de solides, en terme de qualité d'approximation de la distance Euclidienne pour le solide résultant et en tant que continuité de la fonction définissant le solide [23]. Les objets définis en utilisant les R-fonctions ont tendance a perdre rapidement la qualité d'approximation de la distance pour les lignes de champ n'étant pas aux alentours de la surface du solide. Les objets définis en utilisant min/max pour définir les opérations utilisées pour la construction du solide proposent une meilleure approximation de la distance Euclidienne mais introduisent des points de discontinuité des dérivés partielles en chaque ligne de champs. Les opérations SARDF proposent une meilleure approximation de la distance Euclidienne que les R-fonctions et min/max et n'introduisent pas de disconti-

nuités supplémentaires des dérivées partielles [23]. La figure 3.2 illustre les opérations SARDF comparées aux R-fonctions et á min/max dans le cas de l'union de deux ellipsoides.

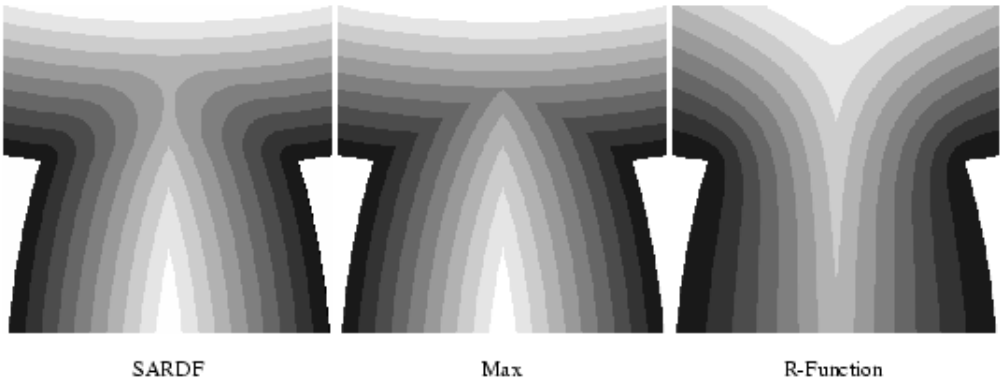


Figure 3.2 — Lignes de contour de l'union de deux ellipsoides avec différentes définitions de l'union. Le niveau de gris correspond à la distance de points à l'intérieur du solide vers la surface. Gauche : SARDF union, milieu : max, les corners indiquent la discontinuité C^1, droite : R-union, les lignes de contour indique clairement que la fonction perd la notion de distance même proche de la surface.

3.1.5 Applications

Nous présentons ensuite une application du framework SARDF pour la modélisation d'objets hétérogènes [23]. Nous proposons une extension du modèle constructif hypervolume [51], dans lequel les opérations utilisées sont des opérations SARDF et les primitives sont définis par la distance Euclidienne.

Les opérations SARDF sont utilisées à la place des R-fonctions ou de min/max dans les différents arbres constructifs utilisés pour définir la géométrie du solide ainsi que la géométrie des partitions où les attributs sont définis.

Nous illustrons à l'aide de modèles en deux dimensions (voir figure 3.3) et en trois dimensions (voir figure 3.4) les champs de distances au solide et partitions ainsi que l'effet sur la définition et la modélisation des attributs. Nous montrons que l'utilisation du framework SARDF facilite la création de modèles hétérogènes et le contrôle de la modélisation des attributs.

Cette approche est une solution possible à la question posée dans [8] sur les méthodes pouvant être utilisées pour la construction de champs de distance utilisés pour paramétriser les distributions de matériaux.

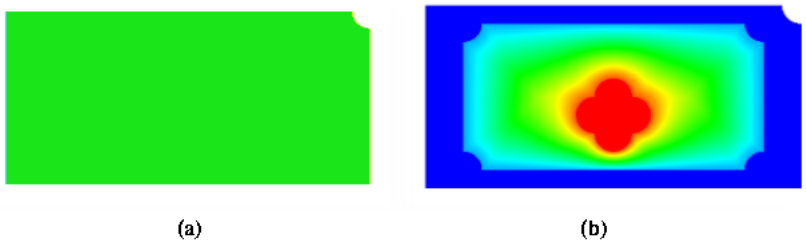


Figure 3.3 — Exemple d'un objet modélisé avec les SARDF en deux dimensions : (a) Géométrie de la pièce. (b) Régions matérielles (bleu : matériel 1, rouge : matériel 2, gradient de couleur : matériel défini par une interpolation des matériaux précédents).

3.2 Modèle FRep template et optimisation de FRep

3.2.1 Objectifs

La modélisation constructive d'objets est difficile et requiert beaucoup de temps et d'efforts. Afin de simplifier et d'automatiser ce processus, nous proposons l'utilisation de modèles FRep templates [21, 20, 24]. Nous considérons des familles paramétriques de solides définies par des FRep ou avec le framework SARDF où les paramètres sont utilisés pour contrôler la forme globale du solide. En effet, différents objets peuvent être regroupés en familles d'objets et présenter de nombreuses similitudes, comme par exemple différents vases.

Nous considérons les problèmes d'optimisation d'objets FRep templates pour la reconstruction d'objets définis par des nuages de points, ainsi que les problèmes de remaillage de BRep.

3.2.2 Algorithmes

L'optimisation des paramètres d'un modèle FRep template pour la reconstruction de solides représentés par des nuages de points est réalisée par différents types d'algorithmes.

Le modèle FRep F définit la surface d'un solide par l'ensemble des points P pour lesquels $F(P) = 0$. Par conséquent, optimiser les paramètres a_i d'un modèle template FRep F pour que le modèle définisse un solide correspondant à un nuage de points consiste à trouver le vecteur $A = (a_i)$, $i = 1..n$ tels que : en chacun des points P_j, $j = 1..m$ du nuage de point, $F(P_j, A) = 0$. Cela revient a minimiser la fonction $\sum_j F^2(P_j, A)$. Il s'agit d'un problème de minimisation de fonction non-linéaire.

Nous proposons et étudions différents algorithmes basés sur des métaheuristiques

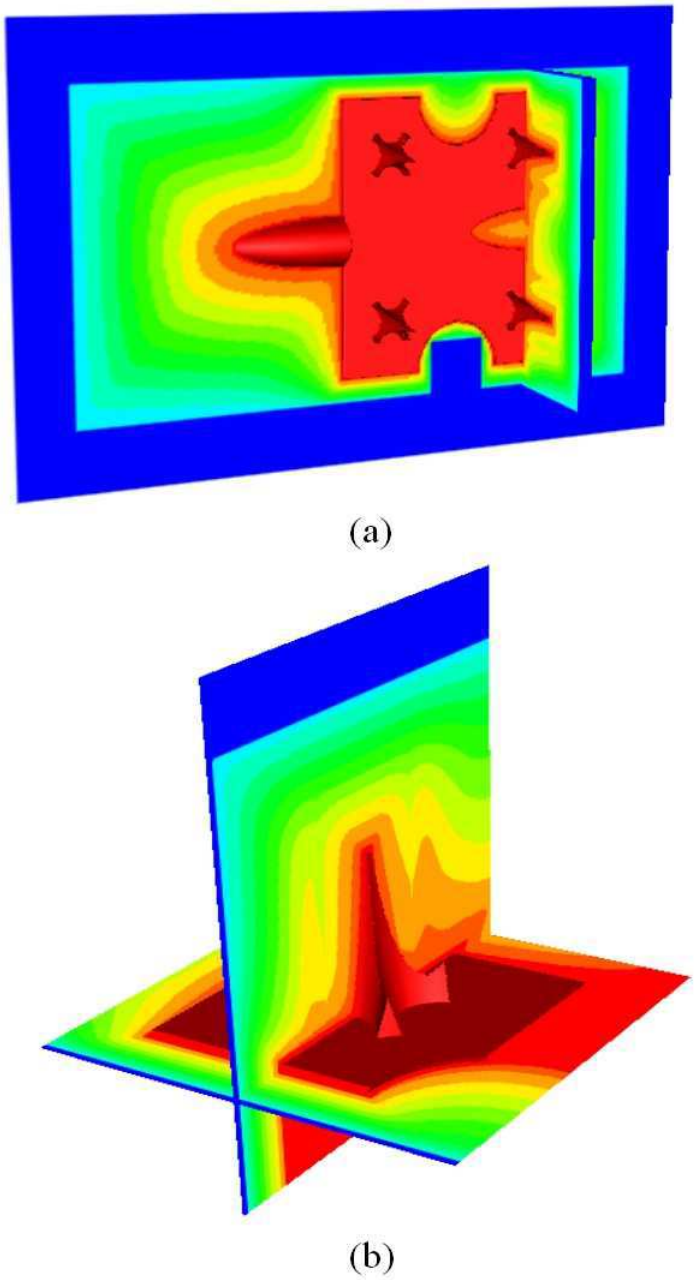


Figure 3.4 — Distribution de deux matériaux pour un objet en trois dimension. La couleur bleue correspond au matériau 1, la couleur rouge au matériau 2. Le gradient de couleur correspond à la fraction de matière. (a) Deux cross-sections correspondent à $x = 0$ et $y = 0$ et montrent les distributions de matières. (b) Un zoom est fait sur un détail de l'objet.

telles que le recuit simulé [21, 20] ou encore des algorithmes génétiques [24] pour résoudre ce problème.

Ces algorithmes sont combinés avec des algorithmes de type Gauss-Newton ou Levenberg Marquardt pour améliorer la vitesse de convergence et la robustesse de la méthode [21].

Des algorithmes comme le recuit simulé ou les algorithmes génétiques sont généralement capables d'éviter de converger ver un minimum local, à l'opposé d'algorithmes comme Gauss-Newton ou Levenberg Marquardt. La figure 3.5 illustre l'utilisation de l'algorithme de Levenberg Marquardt pour l'optimisation d'un pot de saké et l'effet de minimum local. Pour cela il est nécessaire de commencer l'optimisation par un algorithme reposant sur des méta-heuristiques. L'algorithme de Levenberg Marquardt ou Gauss-Newton peut être utilisé en combinaison avec l'algorithme de recuit simulé ou un algorithme génétique pour améliorer la vitesse de convergence et améliorer la robustesse du résultat.

Figure 3.5 — Effet de minimun local : le résultat de l'optimisation avec une méthode d'optimisation locale d'un modèle de pot de saké. Le modèle original et le modèle optimisé sont montrés ensemble pour comparaison.

3.2.3 Résultats et exemples

Nous illustrons ces algorithmes pour la reconstruction de différents objets définis par des nuages de points, comme par exemple un pot de saké (voir figure 3.6).

Nous étudions aussi l'utilisation des algorithmes précédents pour la reconstruction de FRep à partir de BRep [20], et le remaillage du modèle FRep avec la méthode

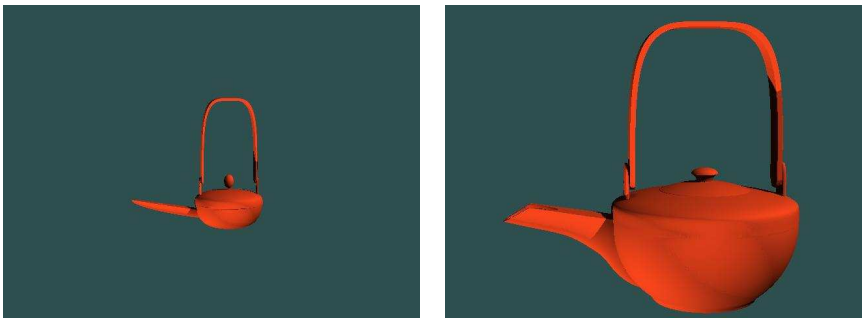


Figure 3.6 — Evolution des formes d'un pot de saké pendant son optimisation à partir d'un nuage de points

de propagation de front [37]. Le remaillage de surface est très important pour de nombreuses applications d'analyse numérique comme par exemple, les méthodes de calcul d'éléments finis.

L'utilisation de modèle FRep template repose sur le fait que de nombreux modèles dans un domaine d'application présente des formes communes, comme par exemple des vases, des pièces en ingénierie mécanique. La question qui se pose est comment automatiser la création d'un modèle template, ou comment induire un modèle FRep constructif à partir d'un nuage de points. Nous proposons des solutions à ce problème dans la suite.

3.3 Création de modèles FRep constructifs

Nous proposons différents algorithmes pour reconstruire un modèle FRep défini par un arbre constructif à partir d'un nuage de points.

3.3.1 Algorithmes pour la reconstruction de modèle FRep constructif à partir de nuage de points

Le premier algorithme [22] reconstruit un modèle FRep constructif à partir d'un nuage de points segmentés et d'une liste de primitives associées à chaque sous-ensemble du nuage de points original. L'algorithme est basé sur un algorithme génétique ; chaque individu de la population représente une solution du problème et encode un modèle FRep en une suite de gènes. Le meilleur individu sélectionné par l'algorithme génétique correspond à la meilleure combinaison de primitives et d'opérations entre ces primitives qui représente au mieux le solide qui doit être reconstruit. La fonction utilisée pour classer les différents individus au sein d'une population de possibles solutions obtenues au cours d'une itération de l'algorithme génétique est : $\sum_i F_j(P_i)$, où F_j est la FRep obtenue en décodant l'individu j de la population.

Cet algorithme ressemble à un algorithme de programmation génétique, où un programme est créé par évolution génétique. La raison pour laquelle un algorithme génétique a été utilisé dans ce cas est que la taille d'une solution au problème est fixe, puisque le nombre de primitives requis pour reconstruire le solide est connu.

Une extension de cet algorithme [72] consiste en l'utilisation de multiples algorithmes génétiques qui vont construire itérativement un arbre pour le modèle FRep en choisissant à chaque itération de l'algorithme la meilleure primitive et opération pour la reconstruction du solide. Chaque itération de l'algorithme correspond à la création d'un niveau de l'arbre constructif définissant le solide. Cette méthode est relativement coûteuse en temps de calcul, mais ne nécessite aucune segmentation préalable du solide.

Le dernier algorithme proposé [72] est basé sur de la programmation génétique et ne nécessite pas de segmentation du nuage de points ou de primitives optimisées pour les différents sous-ensembles du nuage de points, puisque ces étapes sont incorporées dans l'algorithme. Le principe de l'algorithme est d'utiliser la programmation génétique pour créer un programme qui définit un objet géométrique par un arbre constructif. Le principal problème des algorithmes basés sur la programmation génétique vient du fait que le type de donnée utilisé, un arbre, est de taille potentiellement infinie. En effet, l'arbre peut grandir indéfiniment, en incorporant des sous-arbres n'apportant aucune information supplémentaire au problème ; par exemple dans le cas de la modélisation par FRep, des expressions comme $f \vee f$ (l'union du solide représenté par f avec lui-même), vont accroître la taille de la représentation, sans pour autant apporter des informations supplémentaires nécessaires pour la résolution du problème. La programmation génétique permet la construction automatique d'un modèle FRep constructif à partir d'un nuage de points, malheureusement, l'expression finale représentant le solide est généralement plus longue et compliquée qu'elle ne devrait. Ce type d'algorithme est aussi coûteux en temps de calcul.

CHAPITRE 4

Conclusion

4.1 Contributions principales et limitations

Dans ce document, un framework pour la construction d'objets volumétriques utilisant des champs de distance basés sur la distance Euclidienne a été présenté. Ce framework est appelé SARDF (pour Signed Approximate Real Distance Function), et forme un sous-ensemble du modèle FRep [52]. Dans ce framework, un objet est modélisé de manière constructive, en appliquant des opérations (les opérations SARDF) à des primitives (définies par la distance Euclidienne ou son approximation). Les primitives et opérations sont définies par des fonctions réelles, avec la propriété suivante : l'objet final, obtenu en appliquant des opérations aux fonctions (correspondant aux primitives) correspond encore à une fonction. Nous avons ensuite présenté des algorithmes pour faciliter, et automatiser la modélisation d'objets définis dans ce framework. Nous rappelons plus en détails les principaux résultats de cette thèse :

Les opérations SARDF Nous imposons des critères supplémentaires lors de la modélisation d'objets à l'intérieur du modèle FRep. La fonction finale définissant l'objet doit correspondre à une approximation de la distance Euclidienne signée et doit être suffisament lisse. Les implémentations des opérations ensemblistes, R-functions et min/max, souffrent soit d'une mauvaise approximation de la distance Euclidienne (R-functions), soit ne sont pas suffisament lisses et créent des discontinuités des dérivées partielles de la fonction construite (min/max). Nous avons introduit de nouvelles fonctions, et présenté leurs constructions et implémentations pour définir des opérations ensemblistes - intersection, union et différences - appelées opérations SARDF, qui forment une approximation raisonable de la fonction distance Euclidienne et sont lisses. Nous avons prouvé que les opérations SARDF sont des fonctions C^1.

Les opérations SARDF ne correspondent pas exactement à la distance Euclidienne, à l'exception de certains cas simples, mais introduisent une erreur. Il semble par ailleurs difficile de calculer l'erreur maximale de manière générale. Nous avons comparés pour différents types d'objets le résultat obtenu en utilisant les opérations SARDF et des primitives définies par la distance Euclidienne avec la distance à un maillage correspon-

dant à l'objet précédent et vérifié que la différence entre les deux est toujours faible. Notons cependant que la distance à l'objet calculée par la distance à un maillage de l'objet ne correspond pas non plus exactement à la distance Euclidienne du fait de l'approximation faite lors du maillage de la surface de l'objet et de l'approximation faite lors du calcul de la distance au maillage.

Définition de primitives Pour que la fonction finale définissant l'objet se comporte comme la fonction distance Euclidienne (ou son approximation), les primitives utilisées lors de la construction doivent définir des champs de distance Euclidienne (ou son approximation). La construction de la distance signée a été détaillée pour de nombreuses primitives allant des quadriques à des formes libres plus compliquées.

Le framework SARDF Le framework SARDF a été implémenté et présenté. L'interpréteur du projet HyperFun a été modifié et amélioré pour permettre la modélisation d'objets avec le framework SARDF. Une nouvelle version de l'applet HyperFun a été présentée qui permet de choisir d'utiliser le framework SARDF ou les R-fonctions pour la modélisation d'objets. Dans le premier cas, une implémentation en language Java des opérations SARDF est utilisée avec les autres opérations et primitives autorisées dans le cadre du framework SARDF. Dans le second cas, l'applet fonctionne en mode normal, i.e. elle utilise les R-fonctions ou min/max ainsi que les primitives et opérations usuelles de la bibliothèque HyperFun.

Modélisation d'objets hétérogènes constructifs avec SARDF Nous avons proposé l'utilisation du framework SARDF pour la modélisation d'objets hétérogènes en utilisant une extension du modèle hypervolume constructif de Pasko [51]. La fonction distance est utilisée dans ce cas pour paramétriser la distribution de matériau [8]. Nous avons illustré par de nombreux exemples, les propriétés des SARDF pour la modélisation d'objets hétérogènes.

Le framework SARDF supporte la modélisation constructive de solides, qui peut être une tâche répétitive et difficile. Afin de faciliter et d'automatiser ce processus, nous avons proposé d'utiliser des modèles FRep paramétriques et proposé des algorithmes pour optimiser ces objets paramétriques en fonction de diverses contraintes.

Modèles paramétriques et optimisation Il est possible de regrouper plusieurs objets ayant des formes similaires au sein de familles d'objets : par exemple, différents vases possèdent une structure et forme similaire. Une fois qu'un modèle a été obtenu pour un objet, différents paramètres correspondant aux caractéristiques de l'objet peuvent être identifiés. Ces paramètres sont utilisés par la suite pour adapter le modèle

à différentes instances de l'objet. Pour reprendre l'exemple du vase : des vases peuvent avoir la même forme mais différentes largeurs.

L'utilisation de modèles paramétriques et de différents algorithmes pour optimiser leurs formes en fonction de différentes contraintes a été illustrée avec différents exemples de modèles paramétriques pour des pièces mécaniques ainsi qu'un pot à de saké. Un autre exemple de pièce mécanique a été utilisé pour illustrer l'utilisation de l'optimisation de modèles paramétriques dans des applications de remaillage pour des méthodes aux éléments finis.

Le principal inconvénient de cette méthode est la nécessité de construire un modèle paramétrique pour chaque famille de solide. Les algorithmes pour optimiser les paramètres sont aussi coûteux en temps de calcul.

Reconstruction de modèles constructifs Un nouvel algorithme pour la reconstruction de modèles FRep ou SARDF constructifs à partir de nuages de points segmentés a été présenté. Étant donnée une liste des primitives associées au nuage de points segmenté, l'algorithme proposé cherche l'expression ensembliste utilisant les primitives données qui va décrire au mieux le solide. L'algorithme proposé n'est pas limité aux opérations ensemblistes et peut être étendu avec les opérations de blending, par exemple.

L'inconvénient de cette méthode est la nécessité de segmenter le nuage de points et d'associer une primitive à chaque sous-ensemble de points.

Une approche différente de l'algorithme précédent, utilisant la programmation génétique a été proposée. Contrairement à l'approche précédente, cette méthode ne nécessite pas que le nuage de points soit segmenté. Cette méthode est par contre plus exigeante en terme de temps CPU.

A partir du modèle constructif FRep ou SARDF reconstruit, il est possible de créer un modèle paramétrique, qui peut être reutilisé pour l'optimisation de formes en fonction de contraintes, l'optimisation de maillages ou d'autres applications.

4.2 Extensions possibles

Le processus de modélisation peut être grandement automatisé par la réutilisation de modèles paramétriques et leurs optimisations. L'utilisation de modèles paramétriques peut être étendue dans plusieurs directions : dans ce document, nous avons limité notre attention à l'optimisation de formes minimisant la distance vers un nuage

de points ; mais différent types d'optimisation peuvent être envisagés.

L'idée de modèles paramétriques FRep conduit vers l'idée plus générale de paramétrisation des attributs et l'automatisation de la modélisation des propriétés (attributs). Comme pour la paramétrisation de la géométrie du solide, la géométrie associé aux attributs peut être paramétrisée et optimisée pour satisfaire des contraintes de modélisation. L'utilisation du framework SARDF pour définir des modèles permet l'utilisation de la distance pour la modélisation de la forme et des propriétés comme un paramètre naturel pour l'optimisation.

En limitant les attributs à des propriétés surfaciques de l'objet, il est possible de paramétriser et reconstruire la texture des objets. Un modèle template peut être défini pour la géométrie de l'objet et un modèle template peut être défini pour la texture de l'objet. Ces deux modèles peuvent ensuite être ré-utilisés et optimisés pour différentes instances d'objets.

La création de modèles paramétriques et certains algorithmes de reconstruction nécessite l'existence d'une segmentation du nuage de points original et l'association de primitives á chaque sous-ensemble de points. Le problème de la segmentation de nuage de points n'a pas été abordé dans ce manuscrit mais contient de nombreuses directions de recherche.

Le problème de l'association de primitives à chaque sous-ensemble du nuage segmenté contient aussi de nombreuses pistes de recherche. Ce problème et le problème précédent sont liés. Certaines méthodes combinent la recherche de primitives et la segmentation alors que d'autres méthodes construisent une segmentation du nuage de points en premier et optimisent ensuite des primitives pour chaque sous-ensembles.

Publications personnelles

Contribution à un ouvrage

– P.-A. Fayolle, A. Pasko, E. Kartasheva, C. Rosenberger, C. Toinard, Automation of the Volumetric Models Construction, in Heterogeneous Objects Modeling and Applications, Lecture Notes in Computer Science, vol. 4889, A. Pasko, V. Adzhiev, and P. Comninos (Eds.), Springer-Verlag, pp. 214-238, 2008.

Revues Internationales

– P.-A. Fayolle, A. Pasko, B. Schmitt, and N. Mirenkov, Construc- tive heterogeneous object modeling using signed approximate real distance functions, Journal of Computing and Information Science in Engineering, Transactions of the ASME, 6 (2006), no. 3,
– S. Silva, P.-A. Fayolle, J. Vincent, G. Pauron, C. Rosenberger, C. Toinard, "Genetic Algorithms For Shape Modelling and Fitting" Lecture Notes in Computer Science, ISBN : 3-540-30737-0, vol 3808, pages 144-155, 2005 (selectionné du workshop ALEA de la Conference EPIA).
– P.-A. Fayolle, B. Schmitt, Y. Goto, and A. Pasko, Web-based constructive shape modeling using real distance functions, IEICE Transactions on Information and System (2005), no. 5, 828-835.
– P.-A. Fayolle, A. Pasko, and N. Mirenkov, Fitting of parameterized frep shape models, The Journal of Three Dimensional Images (2004), no. 1, 40-46.
– C. Vilbrandt, G. Pasko, A. Pasko, P.-A. Fayolle, T. Vilbrandt, J. Goodwin, J. Goodwin, and T. Kunii, Cultural heritage preservation using constructive shape modeling, Comp. Graph. Forum (2004), no. 1, 25-41.

Conférences Internationales

– P.-A. Fayolle, A. Pasko, E. Kartasheva, and N. Mirenkov, Shape recovery using

functionally represented constructive models, Pro- ceedings of International Conference on Shape Modeling and Ap- plications 2004 (SMI'04), 2004, pp. 375-378.

– P.-A. Fayolle, C. Rosenberger, C. Toinard "Shape Recovery and Functional Modeling Using Genetic Algorithms" IEEE Virtual Reality International Conference (VRIC), pages 227-232, Laval, 2004.
– P.-A. Fayolle, C. Rosenberger, and C. Toinard, 3d shape recon- struction of template models using genetic algorithms, Proceedings of 17th International Conference on Pattern Recognition (ICPR'04), 2004, pp. 269-272.
– P.-A. Fayolle, A. Pasko, N.Mirenkov, C. Rosenberger, and C. Toinard, Constructive tree recovery using genetic algorithms, Procedings of the International Conference on Visualization, Imaging and Image Processing, 2006.
– P.-A. Fayolle, S.Silva, G. Latinier, D. Saffrey, C. Rosenberger, C. Toinard, "Shape Modeling With Genetic Programming", IEEE Virtual Reality International Conference, (VRIC), pages 235-241, 2006.

Bibliographie

[1] N. Amenta, S. Choi, and R. K. Kolluri. The power crust. In *SMA '01 : Proceedings of the sixth ACM symposium on Solid modeling and applications*, pages 249–266, New York, NY, USA, 2001. ACM Press.

[2] L. Barthe, N. A. Dodgson, M. A. Sabin, B. Wyvill, and V. Gaildrat. Two-dimensional potential fields for advanced implicit modeling operators. *Computer Graphics Forum*, 22(1) :23–33, 2003.

[3] B. G. Batchelor. Hierarchical shape description based upon convex hulls of concavities. *Journal of Cybernetics*, 10 :205–210, 1980.

[4] P. Benko, G. Kos, T. Varady, L. Andor, and R. Martin. Constrained fitting in reverse engineering. *Computer Aided Geometric Design*, 19(3) :173–205, 2002.

[5] P. Benko and T. Varady. Direct segmentation of smooth, multiple point regions. In *Proceedings of GMP*, pages 169–178, 2002.

[6] S. Bhashyam, K.H. Shin, and D. Dutta. An integrated cad system for design of heterogeneous objects. *Rapid Prototyping Journal*, 6(2) :119–135, 2000.

[7] A. Biswas and V. Shapiro. Approximate distance fields with non-vanishing gradients. *Graph. Models*, 66(3) :133–159, 2004.

[8] A. Biswas, V. Shapiro, and I. Tsukanov. Heterogeneous material modeling with distance fields. *Comput. Aided Geom. Des.*, 21(3) :215–242, 2004.

[9] J. Blinn. A generalization of algebraic surface drawing. *ACM Trans. Graph.*, 1(3) :235–256, 1982.

[10] J. Bloomenthal. Polygonization of implicit surfaces. *Computer Aided Geometric Design*, 5(4) :341–355, 1988.

[11] J. Bloomenthal, C. Bajaj, J. Blinn, M.-P. Cani-Gascuel, A. Rockwood, B. Wyvill, and G. Wyvill. *Introduction to implicit surfaces*. Morgan-Kaufmann, 1997.

[12] A. Bowyer. *SvLis – Introduction and User Manual*. Information Geometers, Ltd, 1995.

[13] D. E. Breen and R. T. Whitaker. A level-set approach for the metamorphosis of solid models. *IEEE Transactions on Visualization and Computer Graphics*, 7(2) :173–192, 2001.

[14] P. Brunet and I. Navazo. Solid representation and operation using extended octrees. *ACM Trans. Graph.*, 9(2) :170–197, 1990.

[15] S. F. Buchele and R. H. Crawford. Three-dimensional halfspace constructive solid geometry tree construction from implicit boundary representations. *Computer-Aided Design*, 36(11) :1063–1073, 2004.

[16] F. Cazals and M. Pouget. Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design*, 22(2) :121–146, 2005.

[17] K. Chen and X. Feng. Computer-aided design method for the components made of heterogeneous materials. *Computer-Aided Design*, 35(5) :453–466, 2003.

[18] L. Chevalier, F. Jaillet, and A. Baskurt. Segmentation and superquadric modeling of 3d objects. In *Proceedings of WSCG 2003.*, 2003.

[19] D. Dobkin, L. Guibas, J. Hershberger, and J. Snoeynik. An efficient algorithm for finding the csg representation of a simple polygon. *Computer Graphics*, 22(4) :31–40, 1988.

[20] P.-A. Fayolle, A. Pasko, E. Kartasheva, and N. Mirenkov. Shape recovery using functionally represented constructive models. In *Proceedings of International Conference on Shape Modeling and Applications 2004 (SMI'04)*, pages 375–378, 2004.

[21] P.-A. Fayolle, A. Pasko, and N. Mirenkov. Fitting of parameterized frep shape models. *The Journal of Three Dimensional Images*, 18(1) :40–46, 2004.

[22] P.-A. Fayolle, A. Pasko, N. Mirenkov, C. Rosenberger, and C. Toinard. Constructive tree recovery using genetic algorithms. In *Procedings of the International Conference on Visualization, Imaging and Image Processing*, 2006.

[23] P.-A. Fayolle, A. Pasko, B. Schmitt, and N. Mirenkov. Constructive heterogeneous object modeling using signed approximate real distance functions. *Journal of Computing and Information Science in Engineering, Transactions of the ASME*, 6(3), September 2006.

[24] P.-A. Fayolle, C. Rosenberger, and C. Toinard. 3d shape reconstruction of template models using genetic algorithms. In *Proceedings of 17th International Conference on Pattern Recognition (ICPR'04)*, pages 269–272, 2004.

[25] P.-A. Fayolle, B. Schmitt, Y. Goto, and A. Pasko. Web-based constructive shape modeling using real distance functions. *IEICE Transactions on Information and System*, E88-D(5) :828–835, May 2005.

[26] J. D. Foley, A. Vand Dam, S. K. Feiner, and J. F. Hugues. *Computer Graphics : Principles and Practices*. Addison-Wesley, 1995.

[27] M. Freytag, V. Shapiro, and I. Tsukanov. Field modeling with sampled distances. *Computer Aided Design*, 38(2) :87–100, february 2006.

[28] S. F. Frisken, R. N. Perry, A. P. Rockwood, and T. R. Jones. Adaptively sampled distance fields : a general representation of shape for computer graphics. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 249–254. ACM Press/Addison-Wesley Publishing Co., 2000.

[29] R.N. Goldman. Two approaches to a computer model for quadric surfaces. *IEEE Computer Graphics and Applications*, 3(6) :21–24, 1983.

[30] J. Hart. Sphere tracing : A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer*, 12(10) :527–545, 1996.

[31] John C. Hart. Distance to an ellipsoid. In Paul Heckbert, editor, *Graphics Gems IV*, pages 113–119. Academic Press, Boston, 1994.

[32] K. Hoff, T. Culver, J. Keyser, M. Lin, and D. Manocha. Fast computation of generalized voronoi diagrams using graphics hardware. In *Proceedings of ACM SIGGRAPH*, pages 277–286. ACM, 1999.

[33] H. Hoppe. *Surface reconstruction from unorganized points*. PhD thesis, University of Washington, June 1994.

[34] P.-C. Hsu and C. Lee. The scale method for blending operations in functionally-based constructive geometry. *Computer Graphics Forum*, 22(2) :143–158, 2003.

[35] T.R. Jackson. *Analysis of functionally graded material object representation methods*. PhD thesis, MIT, Ocean Engineering Department, 2000.

[36] M. Jones and M. Chen. A new approach to the construction of surfaces from contour data. *Computer Graphics Forum*, 13(3) :75–84, 1994.

[37] E. Kartasheva, V. Adzhiev, A. Pasko, O. Fryazinov, and V. Gasilov. Surface and volume discretization of functionally based heterogeneous objects. *Journal of Computing and Information Science in Engineering, Transactions of the ASME*, 3(4) :285–294, 2003.

[38] V. Kumar, D. Burns, D. Dutta, and C. Hoffman. A framework for object modeling. *Computer-Aided Design*, 31(9) :541–546, 1999.

[39] V. Kumar and D. Dutta. An approach to modeling multi-material objects. In *Proceedings of the Fourth Symposium on Solid Modeling and Applications*, pages 336–345. ACM SIGGRAPH, 1997.

[40] H. Liu, W. Cho, T.R. Jackson, N.M. Patrikalakis, and E.M. Sachs. Algorithms for design and interrogation of functionally gradient material objects. In *Proceedings of ASME 2000 IDETC/CIE 2000 ASME Design Automation Conference, Baltimore, MD*, 2000.

[41] W.E. Lorensen and H.E. Cline. Marching cubes : a high presolution 3d surface construction algorithm. *Computer Graphics*, 21(4) :163–169, 1987. Proceedings of SIGGRAPH 87.

[42] D. Marshall, G. Lukacs, and R. Martin. Robust segmentation of primitives from range data in the presence of geometry degeneracy. *IEEE Transactions on pattern analysis and machine intelligence*, 23(3), march 2001.

[43] S. Mauch. *Efficient Algorithms for Solving Static Hamilton-Jacobi Equations*. PhD thesis, California Institute of Technology, 2003.

[44] J. McCormack and A. Sherstyuk. Creating and rendering convolution surfaces. *Computer Graphics Forum*, 17(2) :113–120, 1998.

[45] B. Morse, T. Yoo, D. Chen, P. Rheingans, and K. Subramanian. Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. In *Proceedings of Shape modeling international*, pages 89–98, 2001.

[46] S. Muraki. Volumetric shape description of range data using "blobby model". In *Proceedings of SIGGRAPH*, pages 227–235, 1991.

[47] G. Nielson. Volume modelling. *Volume Graphics, M. Chen, A. Kaufman, R. Yagel (Eds.), Springer-Verlag*, pages 29–48, 2000.

[48] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H.-P. Seidel. Multi-level partition of unity implicits. *ACM Trans. Graph.*, 22(3) :463–470, 2003.

[49] Y. Ohtake, A. Belyaev, and H.-P. Seidel. Ridge-valley lines on meshes via implicit surface fitting. *ACM Trans. Graph.*, 23(3) :609–612, 2004.

[50] Y. Ohtake, A. Belyaev, and H.-P. Seidel. An integrating approach to meshing scattered point data. In *SPM '05 : Proceedings of the 2005 ACM symposium on Solid and physical modeling*, pages 61–69, New York, NY, USA, 2005. ACM Press.

[51] A. Pasko, V. Adzhiev, B. Schmitt, and C. Schlick. Constructive hypervolume modeling. *Graphical Models*, 63(6) :413–442, 2001. Special issue in volume modeling.

[52] A. Pasko, V. Adzhiev, A. Sourin, and V. Savchenko. Function representation in geometric modeling : concept, implementation and applications. *The Visual Computer*, 11(8) :429–446, 1995.

[53] A. Pasko, V. Pilyugin, and V. Pokrovskiy. Geometric modeling in the analysis of trivariate functions. *Computers and Graphics*, 12(3/4) :457–465, 1988.

[54] A. Pasko and V. Savchenko. Blending operations for the functionally based constructive geometry. In *set-theoretic Solid Modeling : Techniques and Applications, CSG 94 Conference Proceedings*, pages 151–161. Information Geometers, 1994.

[55] X. Qian and D. Dutta. Physics based b-spline heterogeneous object modeling. In *DETC and Computers and Information in Engineering Conference*. ASME, September 2001.

[56] A. Requicha. Representations for rigid solids : theory, methods, and systems. *ACM Computing Surveys*, 12(4) :437–464, 1980.

[57] A. A. G. Requicha and H. B. Voelcker. Constructive solid geometry. Technical Report 25, Production Automation Project, University of Rochester, November 1977.

[58] A. Ricci. A constructive geometry for computer graphics. *The Computer Journal*, 16(2) :157–160, 1973.

[59] C. Roessl, F. Zeilfelder, G. Nurnberger, and H.-P. Seidel. Spline approximation of general volumetric data. In *Proceedings of ACM Solid Modeling*. ACM Solid Modeling 2004, 2004.

[60] V. Rvachev. On the analytical description of some geometric objects. 153(4) :765–767, 1963.

[61] V. Rvachev. *Methods of Logic Algebra in Mathematical Physics*. Naukova Dumka, Kiev, 1974. In Russian.

[62] V. Rvachev. *Theory of R-functions and Some Applications*. Naukova Dumka, Kiev, 1982. In Russian.

[63] V. L. Rvachev, L. V. Kurpa, N. G. Sklepus, and L. A. Uchishvili. Method of r-functions in problems on bending and vibrations of plates of complex shape. 1973. in russian.

[64] M. Sabin. The use of potential surfaces for numerical geometry. Technical Report VTO/MS/153, British Aircraft Corporation, 1968.

[65] V. Savchenko, A. Pasko, O. Okunev, and T. Kunii. Function representation of solids reconstructed from scattered surface points and contours. *Comput. Graph. Forum*, 14(4) :181–188, 1995.

[66] J. Sethian. *Level-Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[67] V. Shapiro. Theory of r-functions and applications : A primer. Technical report, Cornell University, November 1988.

[68] V. Shapiro. Real functions for representation of rigid solids. *Computer-Aided Geometric Design*, 11(2), 1994.

[69] V. Shapiro. A convex deficiency tree algorithm for curved polygons. *International Journal of Computational Geometry and Applications*, 11(2) :215–238, 2001.

[70] V. Shapiro. *Handbook of Computer Aided Geometric Design*, chapter Solid Modeling. Elsevier Science Publisher, 2002.

[71] V. Shapiro and D. L. Vossler. Separation for boundary to csg conversion. *ACM Trans. Graph.*, 12(1) :35–55, 1993.

[72] S. Silva, P.-A. Fayolle, J. Vincent, G. Pauron, C. Rosenberger, and C. Toinard. Evolutionary computation approaches for shaped modelling and fitting. In *Proceedings of EPIA*, 2005.

[73] Y.K. Siu and S.T. Tan. Modeling the material grading and structures of heterogeneous objects for layered manufacturing. *Computer-Aided Design*, 34 :705–716, 2002.

[74] A. Sud, A. Otaduy, and D. Manocha. Difi : Fast 3d distance field computation using graphics hardware. *Computer Graphics Forum*, 23(3) :557–566, 2004. Proceedings of Eurographics 2004.

[75] W. Thibault and B. Naylor. Set operations on polyhedra using binary space partitioning trees. *Computer Graphics*, 21(4) :153–162, 1987.

[76] S. B. Tor and A. E. Middleditch. Convex decomposition of simple polygons. *ACM Transactions on Graphics*, 3(4) :244–265, 1984.

[77] Y.R. Tsai. Rapid and accurate computation of the distance function using grids. *J. Comput. Phys.*, 178(1) :175–195, 2002.

[78] G. Turk and J. OBrien. Shape transformation using variational implicit functions. In *Proceedings of SIGGRAPH*, pages 335–342, 1999.

[79] M. Vanco. *A direct approach for the segmentation of unorganized points and recognition of simple algebraic surfaces*. PhD thesis, Technische universitat Chemnitz, 2003.

[80] T. Varady, R. R. Martin, and J. Cox. Reverse engineering of geometric models – an introduction. *Computer Aided Design*, 29(4) :255–268, 1997.

[81] K. Watanabe and A. G. Belyaev. Detection of salient curvature features on polygonal surfaces. *Comput. Graph. Forum*, 20(3), 2001.

[82] J. R. Woodwark and A. L. Wallis. Graphical input to a boolean solid modeller. In *Proc. CAD'82*, pages 681–688, 1982.

[83] H. Zhao. A fast sweeping method for eikonal equations. *Mathematics of Computation*, 2004.

[84] Y. Zhou, A. Kaufman, and A. Toga. 3d skeleton and centerline generation based on an approximate minimum distance field. *The Visual Computer*, 14(7) :303–314, 1998.

PAPER *Special Section on Cyberworlds*

Web-based constructive shape modeling using real distance functions

Pierre-Alain FAYOLLE[†], **Benjamin SCHMITT**[††], **Yuichiro GOTO**[†††], ***and*** **Alexander PASKO**[††††], ***Nonmembers***

SUMMARY We present an approach and a web-based system implementation for modeling shapes using real distance functions. The system consists of an applet supporting the HyperFun modeling language. The applet is extended with primitives defined by Euclidean distance from a point to the surface of the shape. Set-theoretic operations (union, intersection, difference) that provide an approximation of the Euclidean distance to a shape built in a constructive way are introduced. Such operations have a controllable error of the exact Euclidean distance to the shape and preserve C^1 continuity of the overall function, which is an important condition for further operations and applications. The proposed system should help model various shapes, store them in a concise form, and exchange them easily between different entities on a network. The applet offers also the possibility to export the models defined in the HyperFun language to other formats for raytracing or rapid prototyping.

key words: *Shape modeling, Euclidean distance, implicit surfaces, Java applet, computer graphics.*

1. Introduction

The development of networks of computing devices facilitates collaborative works and the exchange of information. In various area ranging from physics, computer graphics, to shape modeling, great attention is paid to defining and processing geometric shape data. An uniform and open format for shapes description is needed for their communication between different entities over a network.

Most of the formats used for the description of shapes rely on polygon meshes, which have two main drawbacks: a lack of expressive power for modeling and the size of the files, typically very large, which is not suitable for exchange and collaborative works through networks.

A possible way to define shapes in a concise way is using signed distance functions. The signed distance function can define a closed surface as a zero value point set, taking positive values inside and negative values outside the surface. Implicit surfaces, see [1] and references therein, and the function representation (FRep) [2] include distance functions, but often mix between algebraic and Euclidean distances.

The Euclidean distance from a point to a given point-set is defined as the minimum distance between this point and any point of the set. A comparison between Euclidean and algebraic distances to quadric surfaces [3] suggests that the Euclidean one is preferable. In the rest of the paper we use the term "distance" to refer to the Euclidean distance.

Distance based models are in fact extremely useful in many applications such as: constant-radius offsetting and blending operations [4], surface metamorphosis and smoothing [5], object reconstruction from a set of cross-sections [6], rendering with sphere tracing [7], generation of skeletal shape representation [8], heterogeneous object modeling [9], and others.

We propose both the theoretical framework and a system implementation for modeling constructive objects defined by signed distance to the surface of the shape. The implemented modeling system is an extension of the HyperFun Java applet for shape modeling through the web [10]. The applet supports the HyperFun language intended for FRep modeling [11], and its core is made by an interpreter for the HyperFun language. The system allows for building complex shapes by using a constructive approach: starting from simple primitives and applying set-theoretic and other operations to them. We introduce new formulations for the set-theoretic operations (union, intersection and differences) that maintain a C^1 continuous (with the exception of some points) approximation of the distance to the surface with a controlled error. Primitives of the system are implemented as part of the library, and the set-theoretic operations as part of the core interpreter.

2. Previous works

2.1 Web-based shape modeling

Several approaches have been considered for shape modeling on the web. For example, Computer Aided Design (CAD) model browsers and converters such as [12], [13], or virtual world authoring tools in Virtual Reality Modeling Language (VRML). Within the VRML framework, some researchers introduced nodes

Manuscript received August 29, 2004.
Manuscript revised November 27, 2004.
Final manuscript received December 10, 2004.

[†]The author is with the University of Aizu, Japan
[††]The author is with Computer Graphics Research Institute and Hosei University (Digital Media Professional), Tokyo, Japan
[†††]The author is with Kanazawa IT, Tokyo, Japan
[††††]The author is with Hosei University, Tokyo, Japan

for defining shapes by skeletal implicit surfaces [14], for visualization of scientific data [15], or visualization of trimmed NURBS [16]. Min et al. [17] introduced a node for FRep shape modeling in VRML, allowing to mix FRep shape models with other valid VRML nodes. However, none of these tools are pure shape modelers. None of them propose an interactive Graphical User Interface (GUI) for editing functionally defined shapes. The work [17] is probably the closest to our approach, but differs in two points: it is oriented to visualization of implicit surfaces embedded in complex VRML scenes rather than to providing a framework for shape modeling; and it is based on algebraic distance functions rather than Euclidean distances.

2.2 Distance function construction

Deriving analytical expressions for a distance function for a given shape may be a tedious work, or even sometimes impossible. Applying constructive modeling [18] is one of the practical solutions to derive a distance function. Constructive modeling is based on applying successively set-theoretic or other operations to predefined shapes (primitives). Let two solids (point sets in Euclidean space) be denoted by $f_1 \geq 0$ and $f_2 \geq 0$, then the union of these two objects is defined by $f_\cup = max(f_1, f_2)$ and the intersection by $f_\cap = min(f_1, f_2)$ [19], [18]. Now, if $f_1(x, y, z)$ and $f_2(x, y, z)$ correspond to the distances from (x, y, z) to the surfaces defined by $f_1 = 0$ and $f_2 = 0$, $f_\cup$ and $f_\cap$ correspond to approximate distance functions for the entire complex object. However, *min / max* operations result in C^1 discontinuity at any point where $f_1 = f_2$. It can cause unexpected results in further operations on the object such as blending, metamorphosis, and others.

There are works trying to replace *min / max* to overcome this C^1 discontinuity: Rvachev proposed the R-functions [20], [21], that provide exact set-theoretic operations, but do not conserve the distance value. R-functions also suffer from exponential value growth when for example applying them to define union of a number of overlapping solids. A normalization [22], [23] can be applied but is computationally expensive and approximates the distance function only close to the surface. The superelliptic approximations of *min / max* [18] do not describe exact set-theoretic operations and suit only for blending. The elliptic approximation of *min / max* by Barthe et al [24] is designed initially for blending and the error of the distance function grows infinitely far from the boundary.

We extend the latter approach by using a circular *min / max* approximation (to remove the C^1 discontinuity) and introduce a bounding band to this approximation in order to limit the error growth and to guarantee a fixed upper error. These set-theoretic operations form the basis of our web based modeling system with the primitives defined by signed real distance functions. Blending operations are derived from the set-theoretic ones, and offsetting and other isometric operations (sometimes also referred as rigid body motions or Euclidean motions) complete them.

3. Distance based primitives and operations

The core of the proposed modeling system is an interpreter for the HyperFun language and a library of available shapes (primitives) and operations (set-theoretic, blending, etc.) [11].

Distance based primitives are implemented in the library and replace the existing algebraic primitives. Then, the core interpreter is modified with the new set theoretic operations allowing a controlled approximation of the distance. Finally, other modeling operations are added to the library: some of them are unchanged compared to the original system, while others are derived from the new set-theoretic operations. This section deals only with mathematical and theoretical details. The details of implementation of the applet are given in section 4.

3.1 Distance based primitives

As mentioned earlier, we are interested only in primitives for which an expression or an evaluation procedure for the Euclidean distance from the current point to the surface is available. Actually, it is possible to have analytical expressions for all the quadrics. It should be noticed that quadrics are the main primitives of most of the constructive solid geometry (CSG) solid modeling kernels, and are enough to model most of the objects at least in mechanical engineering.

Primitives can be classified into two groups:

- primitives with analytical expressions for the distance: cylinder, sphere, torus, cone, block. Expressions for these primitives can be found in [7].
- primitives with procedural distance evaluation: ellipsoid [25], general quadrics.

The last primitive, general quadric, is a generalization of all the types of quadrics. Nevertheless, it is more efficient to use the analytical expressions if available, rather than the expression for the general quadric. The distance to a general quadric, given as $Q(X) = X^T AX + B^T X + c$, can only be computed by a numerical procedure which is slower and less accurate than evaluating a closed form. This numerical computation requires solving a polynomial of degree 6.

3.2 Smooth function approximation for set-theoretic operations

In constructive solid geometry, a complex shape can be built by applying successively operations on primitives.

When working with primitives defined by distance functions, we wish that the result of applying an operation on one or more distance functions, remains a distance function (closure property).

The use of *min* and *max* was introduced in [19] and [18], *min* and *max* correspond exactly to the intersection and union of two solids defined by implicit surfaces. They also conserve an approximation of the distance for the overall solid. It means that if f_1 and f_2 give the distance to the surface $f_i = 0$, where $i = 1, 2$, $f_\cup = max(f_1, f_2)$ and $f_\cap = min(f_1, f_2)$ yield approximate distance to the union and intersection of solids defined by f_1 and f_2. Unfortunately, *min* and *max* have bad differential properties, and as a result $f_\cup$ and $f_\cap$ have C^1 discontinuity for all points such that $f_1 = f_2$, see the sharp corners Fig. 1, left. This bad behavior can cause unexpected results when later operations are applied.

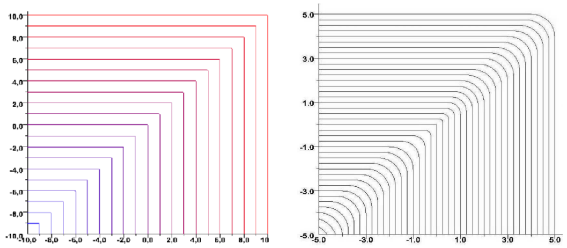


Fig. 1 Contour map $f(x, y) = c$ of the function describing the union of two halfplanes $f_1(x, y) = x \geq 0$ and $f_2(x, y) = y \geq 0$, where $f(x, y) = max(f_1, f_2)$ (left) and SARDF union (right) are applied to define the union.

The idea developed here is inspired by the work of Barthe et al [24] and consists in replacing the sharp corners in the graphs of *min* and *max* (see Fig. 1, left, for the case of the union of two half-planes $f_1(x, y) = x \geq 0$ and $f_2(x, y) = y \geq 0$), by bounded circular approximations (see Fig. 1, right and Fig. 2).

Fig. 2 indicates the main points of the approach for the case of union. The figure corresponds to the contour map of the smoothed union of two half-planes, defined by $f_1(x, y) = x \geq 0$ and $f_2(x, y) = y \geq 0$. At first, a circular arc approximation with growing radius is used to replace any sharp corner (zone I, B). Two straight lines, symmetric with respect to the line $y = x$, are used to delimit the frontier for the circular arc approximation. Notice that because of the growing radius of the approximation, the point at the origin of the (x, y) plane has still a sharp corner. It is needed to keep this sharp corner for modeling exact sharp edges of shapes. The use of circular arc approximation can provide C^1 continuity of the resulting distance function for constructive shapes built using normal primitives (i.e., primitives defined by distance functions). Unfortunately with this approach the radius of the circular arc, used to replace the sharp corners in the contour lines, keeps growing with the distance from the initial surface. Because of this behavior of the arc radius, the error of the distance approximation grows infinitely with the distance.

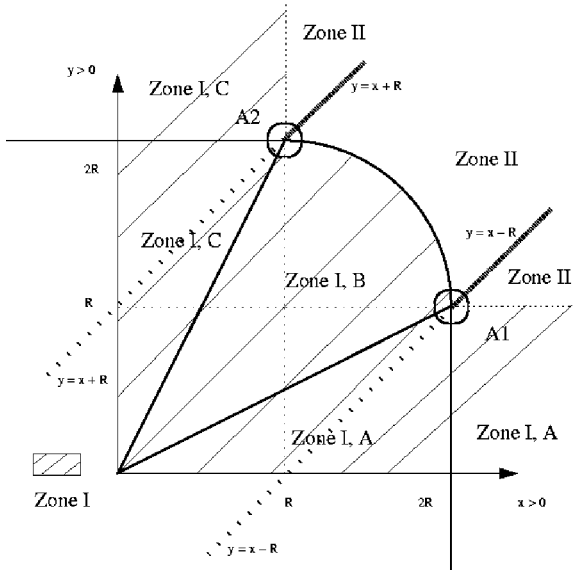


Fig. 2 Contour map $f(x, y) = c$ of the function defining the SARDF union (smoothed union) between two halfplanes $f_1(x, y) = x \geq 0$ and $f_2(x, y) = y \geq 0$. A growing circular arc approximation is applied in zone I, B, whereas we introduce a fixed radius approximation with the bounding band in zone II.

We prevent the radius of the circular arc from growing infinitely by introducing a fixed threshold R. A new bounding band can be introduced by two parallel straight lines that enclose the arcs with the fixed radius (see zone II in Fig. 2). They provide a fixed upper limit of the error at any given point. This approach introduces however a curve of C^1 discontinuity corresponding to the arc of circle boundary between the growing and the fixed radius area. This curve of C^1 discontinuity is localized on one contour line only, and thus does not raise problems in further modeling operations.

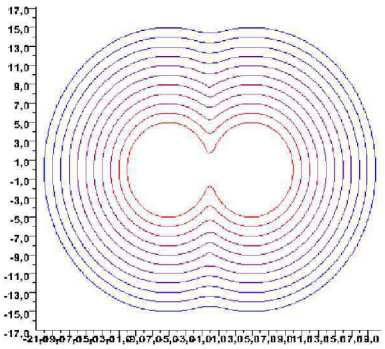


Fig. 3 Contour map $f(x, y) = c$ of the function describing the SARDF union of two disks.

As such distance approximations with limited errors are acceptable for distance based modeling, the resulting defining functions of shapes are called by the term signed approximate real distance functions (SARDF). The union will be called SARDF union and the intersection, SARDF intersection.

While these operations were first derived for the case of two half-planes, they are valid for any shapes. Fig. 3 corresponds, for example, to the contour map of the SARDF union of two disks. The expression of the union between two objects defined by the distance functions f_1 and f_2 is given below as an illustration; it is described by the SARDF union function ($\vee_S$) as follows:

(1) Case 1: $f_1 > 0$ and $f_2 > 0$

In the current paragraph, E_1 is used for the following Boolean expression: $E_1 = (f_1 < R$ or $f_2 < R$ or $(f_1 < 2\ R$ and $f_2 < 2\ R$ and $(f_1 - R)^2 + (f_2 - R)^2 < R^2))$.

$$f_1 \vee_S f_2 = \begin{cases} \frac{-b_1 \pm (b_1^2 - 4a_1c_1)^{0.5}}{2a_1}, \\ \quad \text{if } E_1 \text{ and } \frac{1}{2} < \frac{f_2}{f_1} < 2 \\ f_1, \text{ if } E_1 \text{ and } \frac{f_2}{f_1} \leq \frac{1}{2} \\ f_2, \text{ if } E_1 \text{ and } \frac{f_2}{f_1} \geq 2 \\ \frac{-b_2 \pm (b_2^2 - 4a_2c_2)^{0.5}}{2a_2}, \\ \quad \text{if } !E_1 \text{ and } f_1 - R < f_2 < f_1 + R \\ f_1, \text{ if } !E_1 \text{ and } f_2 \leq f_1 - R \\ f_2, \text{ if } !E_1 \text{ and } f_2 \geq f_1 + R \end{cases}$$

where $a_1 = \frac{1}{4}$, $b_1 = -(f_1 + f_2)$, $c_1 = {f_1}^2 + {f_2}^2$, $a_2 = 2$, $b_2 = -2\ f_1 - 2\ f_2 - 4\ R$ and $c_2 = {f_1}^2 + {f_2}^2 + 2\ f_1\ R + 2\ f_2\ R + R^2$.

(2) Case 2: $f_1 \leq 0$ and $f_2 \geq 0$

$f_1 \vee_S f_2 = f_2$

(3) Case 3: $f_1 < 0$ and $f_2 < 0$

We denote by E_2 the following Boolean expression: $E_2 = (f_1 > -R$ or $f_2 > -R$ or $(f_1 > -2\ R$ and $f_2 > -2\ R$ and $(f_1 + 2\ R)^2 + (f_2 + 2\ R)^2 > R^2))$.

$$f_1 \vee_S f_2 = \begin{cases} -\frac{-b_1 \pm (b_1^2 - 4a_1c_1)^{0.5}}{2a_1}, \\ \quad \text{if } E_2 \text{ and } \frac{1}{2} < \frac{f_2}{f_1} < 2 \\ f_1, \text{ if } E_2 \text{ and } \frac{f_2}{f_1} \leq \frac{1}{2} \\ f_2, \text{ if } E_2 \text{ and } \frac{f_2}{f_1} \geq 2 \\ -\frac{-b_2 \pm (b_2^2 - 4a_2c_2)^{0.5}}{2a_2}, \\ \quad \text{if } !E_2 \text{ and } f_1 - R \leq f_2 \leq f_1 + R \\ f_1, \text{ if } !E_2 \text{ and } f_2 \leq f_1 - R \\ f_2, \text{ if } !E_2 \text{ and } f_2 \geq f_1 + R \end{cases}$$

where $a_1 = 7$, $b_1 = 4\ (f_1 + f_2)$, $c_1 = f_1^2 + f_2^2$, $a_2 = 2$, $b_2 = 2\ f_1 + 2\ f_2 - 4\ R$ and $c_2 = f_1^2 + f_2^2 + 2\ f_1\ R + 2\ f_2\ R + R^2$.

(4) Case 4: $f_1 \geq 0$ and $f_2 \leq 0$

$f_1 \vee_S f_2 = f_1$

The expression of the intersection of two objects defined by distance functions is quite similar by symmetry. The set-theoretic difference between two objects defined by distance functions can be represented as the intersection: $f_1 \backslash_S f_2 = f_1 \wedge_S (-f_2)$.

3.3 Blending operations

It is possible to obtain blending operations with added or subtracted material through small modifications of the SARDF union and intersection operations. The idea consists in using a constant circular arc approximation everywhere to replace every sharp corner in the contour lines of min/max; even the sharp corner at the origin is replaced now by a circular arc. In contrary to the SARDF union and intersection, there are no regions with a growing circular arc approximation (see Zone I,b Fig. 2 for the case of the SARDF union).

These two symmetric blending operations are entirely controlled by a single parameter R, corresponding to the radius of the circular approximation. This parameter defines the shift to be applied to the line $y = x$ in the $\vec{x}$ and $-\vec{x}$ directions, in order to define a bounding band that will enclose the circular arc approximation.

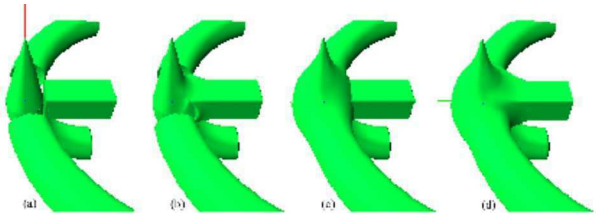


Fig. 4 Blending operations: (a) Initial object composed as a vertical cone, a torus-like shape and a block shape. (b) Blending between the block and the cone. (c) Blending between the torus and the cone. (d) Blending between the block and the cone, and blending between the previous resulting object with the torus.

We propose to illustrate the described blending operations here using an example from section 5.1, and more specifically one of its parts where blending union and blend on blend are used. The area of interest of this shape is shown in Fig. 4 (a). This part is mainly composed of a horizontal block, a cone and some parts of a torus. Two blending union operations are applied in order to blend these three parts. In Fig. 4(b) and (c), we separate each blending function, i.e., respectively a blending union between the cone and the block and a blending union between the cone and the torus. In Fig. 4(d), the resulting object is shown where a blending union is applied between the two blended parts (blend on blend).

4. Applet for distance based shape modeling

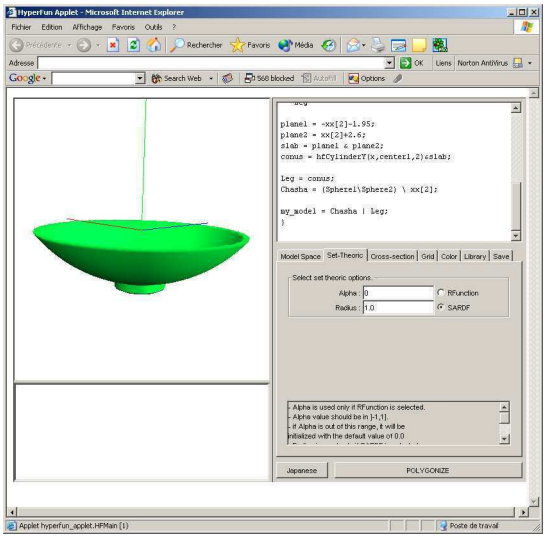


Fig. 5 HyperFun applet for distance based shape modeling.

HyperFun was introduced in [11] as a high-level geometric modeling language fully supporting FRep modeling, and we found it an appropriate basis for an implementation of the herein proposed approach to distance based modeling. Primitives and operations for distance based modeling are implemented within the HyperFun applet described in [10]. At first, we sum up the main points of the applet's components. Then, we explain how the system is modified for distance based shape modeling.

4.1 HyperFun applet components

The applet system is mainly based on three components: a HyperFun to Java bytecode compiler, a polygonizer, and the applet providing the user interface. We briefly characterize the role of each component below.

4.1.1 HyperFun to Java bytecode compiler

The core of the modeling system is an HyperFun to Java bytecode compiler, its architecture is more largely described by R. Cartwright in his PhD dissertation [26], see also [11] and [10]. HyperFun code is translated to calls to the ByteCode Engineering Library (BCEL)† API, resulting in generation of executable Java bytecode. In conjunction with the just-in-time compiler for Java bytecode, it allows for the fast evaluation of defining functions for shape models. The evaluation is needed when rendering the model by raytracing, polygonization, or other methods for rendering FRep objects.

†http://jakarta.apache.org/bcel/

4.1.2 Polygonizer in Java

Polygonization is the process that generates a polygonal approximation of an implicit surface (isosurface). The polygonization used in the applet is based on the algorithm described in [27]. This algorithm falls within a class called exhaustive enumeration as identified by [1]. There are two phases in the algorithm: spatial partitioning and cell polygonization. During spatial partitioning, the rectangular bounding box containing the object is divided into regular cubic cells. The polygonization of cubic cells has topological ambiguities on faces with four edge-surface intersection points. The algorithm in [27] and the one used here resolve the ambiguity by applying a trilinear interpolation in the cell, and a bilinear interpolation on the cell faces. The speed of the algorithm is improved here by using a look-up table similar to the one of the Marching Cubes (MC) algorithm [28]. The look-up table is generated using the algorithm of [27] and is used to resolve the topological ambiguities.

4.1.3 Applet interface

The interface of the applet shown Fig. 5 has four main parts: top left is the rendering window, bottom left is the error message text area, top right is the text area used to input the HyperFun model, and bottom right controls different options.

A model in HyperFun language is input in the dedicated area; pushing the "Polygonize" button compiles the HyperFun code to Java bytecode and calls the polygonizer to generate a polygonal mesh approximation of the object surface. Accuracy of the approximation is controlled by the number of subdivision steps of the bounding box along each axis and can be modified in the menu controls (bottom right). The definition of the bounding box may also be modified in the menus.

The rendering of the polygons is done in the rendering window using the Java3D API. The polygonal mesh can also be exported to a VRML file, to STL format for rapid prototyping, or to a mesh using the PovRay syntax for raytracing.

4.2 Extension for distance-based modeling

In order to support modeling with objects defined by Euclidean distance functions, the core and library of the compiler are modified. At first, the core compiler is modified to allow the use of the SARDF operations (union, intersection, and difference) instead of the min/max or R-functions. SARDF operations as described in 3.2 are implemented. A menu is added to the applet to select the class of operations, i.e., SARDF for distance-based modeling, traditional R-functions, or min/max, and to select the radius threshold R for

the SARDF operations, which controls the upper error bound in the distance approximation. Then, the library is rewritten using primitives with distance functions listed in 3.1 and the blending union and intersection described in 3.3.

The possibility to export the mesh for the PovRay raytracer† is added to the export menu in order to do photo-realistic rendering. Examples of distance based models rendered with PovRay are shown in the following section 5.

5. Examples of distance-based modeling with the applet

After modeling objects using the extended Hyperfun applet, one can export the resulting object either directly as a Hyperfun text file, as an STL file for rapid prototyping, or as a polygonal mesh for further rendering. We propose herein to illustrate this feature with two examples, one related to artistic modeling and rendering, and another to mechanical Computer-Aided Design.

5.1 "CyberVase"

Fig. 6 Polygonal model of a complex artistic vase modeled using the Java HyperFun applet and distance functions and rendered within the applet.

The first example related to artistic modeling is a vase model. Figure 6 shows the polygonal model of the object, modeled and rendered using the HyperFun applet. This object is modeled using the following primitives: ellipsoids, tori, cones and blocks. For the operations, SARDF unions, SARDF intersections, and SARDF blending are used. As one can see, complex objects can be modeled through the applet interface. Although this object could be rendered directly as a legal isosurface object in the raytracing engine PovRay (with adequate library extension), we choose to export this object as a polygonal mesh in order to obtain a photo-realistic rendering of this object. Let us emphasize here the importance of using implicit objects for web-based modeling and as a format for exchange of shapes. The HyperFun file size is less than 5Kb, whereas the polygonal export produces a file larger than 10 Mb (compressed). This large size for the polygonal model is due to, among other factors, the smooth blend of the vase shape that requires a large amount of triangles to approximate properly the original shape. Figure 7 shows the vase rendered with PovRay using transparent material. The vase example illustrates that non-trivial objects can be modeled and that high-quality, photo-realistic rendering can be done on the resulting objects.

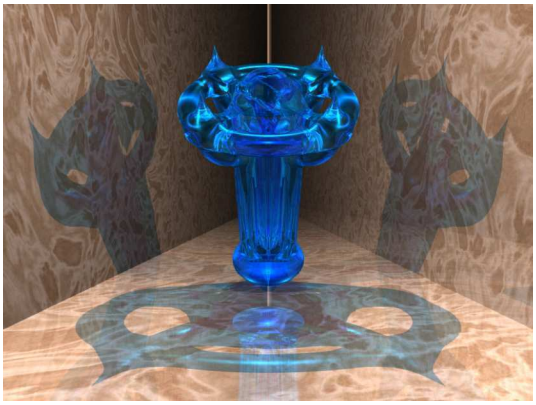


Fig. 7 High quality photo-realistic rendering of the vase modeled using the distance-based HyperFun applet, and rendered using the PovRay raytracer.

5.2 CAD object

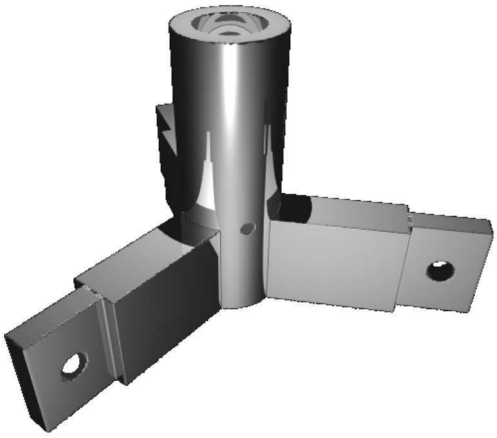


Fig. 8 Raytraced mechanical CAD object modeled using our applet and distance functions.

The second example presented in this section is related to Computer-Aided Design. In this example, a CAD mechanical part is modeled using the HyperFun

†http://www.povray.org

applet. The constructive approach and the distance based modeling allow one to meet the requirements imposed on this shape, i.e., locations of the holes, thickness of the blocks at the end of each branch, and others. The most intriguing result of the distance based modeling approach is the distance field of the object, illustrated in Fig. 9. Each primitive of this object is defined by Euclidean distance, and the set-theoretic operations are defined using SARDF operations. Therefore, the distance field of the object gives an accurate approximation to the exact Euclidean distance function and can be used in further applications such as heterogeneous object modeling [29], [9]. Indeed, to have an Euclidean distance field is very important for modeling material features and material distribution inside the heterogeneous object.

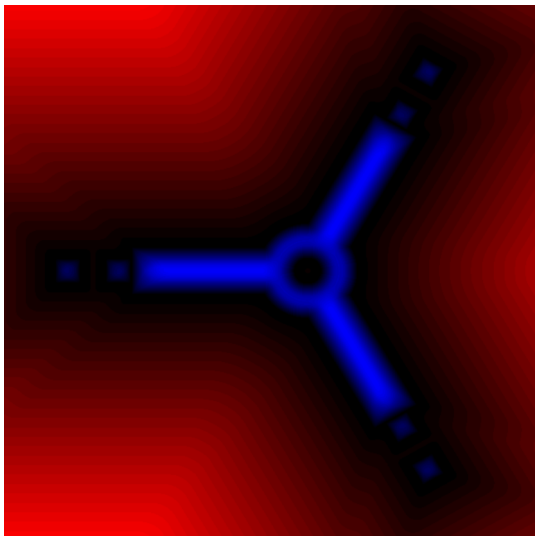


Fig. 9 Distance field of the CAD object at the cross-section by the plane $z = 0$. SARDF operations are defined with the radius of 0.5.

6. Conclusion

Signed Euclidean distance functions describe shapes in a concise way and have many properties that make them highly suitable for shape modeling and its applications. Deriving analytical expressions or implementation of procedures for the Euclidean distance evaluation can be done for some primitives, but is difficult or even impossible for more complex shapes. We resolved the issue by introducing new definitions for set-theoretic operations, called SARDF (for Signed Approximation Real Distance Functions), which can be used for building shapes defined by distance functions in a constructive way with a controllable error. The resulting distance functions for complex shapes are C^1 continuous (with the exception of some points), which is an important condition for further modeling operations. We introduced also new expressions for symmetric blending operations, which can be easily derived from SARDF operations.

The proposed approach has been implemented in an interactive modeling system in the form of a Java applet. The modeling system consists of a subset of the HyperFun language for constructive shape modeling with real Euclidean distance functions. HyperFun language with SARDF operations can serve as a basis of a lightweight protocol for exchanging distance-based shape models over networks. We illustrated through examples from CAD and computer art the key points of the approach: concise way to describe even very complex shapes, bounded approximation of the resulting Euclidean distance function, blending operations. Extending the modeling system to the case of distance-based heterogeneous multi-material objects and for collaborative modeling are the next logical steps in the proposed direction.

Acknowledgement

P.-A. Fayolle acknowledges support by Monbukagakusho, the Japanese Ministry of Education, Culture, Sports, Science and Technology. The authors wish to acknowledge Richard Cartwright for the code of the HyperFun to Java Bytecode compiler, and Mio Hiraga who worked on the HyperFun applet.

References

[1] J. Bloomenthal, C. Bajaj, J. Blinn, M.P. Cani-Gascuel, A. Rockwood, B. Wyvill, and G. Wyvill, Introduction to implicit surfaces, Morgan-Kaufmann, 1997.
[2] A. Pasko, V. Adzhiev, A. Sourin, and V. Savchenko, "Function representation in geometric modeling: concept, implementation and applications," The Visual Computer, vol.11, no.8, pp.429–446, 1995.
[3] R. Goldman, "Two approaches to a computer model for quadric surfaces," IEEE Computer Graphics and Applications, vol.3, no.6, pp.21–24, 1983.
[4] J. Rossignac and A. Requicha, "Constant-radius blending in solid modeling," Computers in Mechanical Engineering, vol.3, no.1, pp.65–73, 1984.
[5] B. Payne and A. Toga, "Distance field manipulation of surface models," IEEE Computer Graphics and Applications, vol.12, no.1, pp.65–71, 1992.
[6] M. Jones and M. Chen, "A new approach to the construction of surfaces from contour data," Computer Graphics Forum, vol.13, no.3, pp.75–84, 1994.
[7] J. Hart, "Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces," The Visual Computer, vol.12, no.10, pp.527–545, 1996.
[8] Y. Zhou, A. Kaufman, and A. Toga, "3d skeleton and centerline generation based on an approximate minimum distance field," The Visual Computer, vol.14, no.7, pp.303–314, 1998.
[9] A. Biswas, V. Shapiro, and I. Tsukanov, "Heterogeneous material modeling with distance fields," Computer-Aided Geometric Design, vol.21, no.3, pp.215–242, 2004.
[10] R. Cartwright, V. Adzhiev, A. Pasko, Y. Goto, and T. Kunii, "Web based shape modeling with hyperfun," Tech. Rep.

HCIS-2002-02, Hosei University, 2002.
[11] A. Adzhiev, R. Cartwright, E. Faussett, A. Ossipov, A. Pasko, and V. Savchenko, "Hyperfun project: a framework for collaborative multidimensional f-rep modeling," pp.59–69, Eurographics / ACM, 1999. Proceeding Implicit Surfaces '99, Eurographics/ACM Workshop, http://www.hyperfun.org.
[12] W. Regli, "Internet-enabled computer aided design," IEEE Internet Computing, vol.1, no.1, pp.39–50, 1997.
[13] R. Oxman and A. Shabo, "The web as a visual design medium," pp.266–271, 1999. International conference on information visualisation, Israel.
[14] B. Wyvill and A. Guy, "The blob tree, implicit modeling and vrml," pp.193–206, 1997. Proceedings Iternational conference from the desktop to the webtop: virtual environments on the internet, WWW and networks.
[15] R. Gimis and D. Nadeau, "Creating vrml extensions to support scientific visualisation," pp.13–20, 1995. Proceeding 1995 Symposium on Virtual Reality Modeling Language.
[16] H. Grahn, T. Volk, and H.J. Wolters, "Nurbs in vrml," pp.35–43, 2000. Proceedings of the Web3D-VRML 2000. Fifth Symposium on Virtual Reality Modeling Language.
[17] L.F. Min, A. Sourin, and K. Levinski, "Function-based 3d web visualization," pp.428–435, 2002. Proceedings of International Symposium Cyber Worlds: Theory and practice 2002.
[18] A. Ricci, "A constructive geometry for computer graphics," The Computer Journal, vol.16, no.2, pp.157–160, 1973.
[19] M. Sabin, "The use of potential surfaces for numerical geometry," Tech. Rep. VTO/MS/153, British Aircraft Corporation, Weybridge, UK, 1968. 13 pages.
[20] V. Rvachev, "On the analytical description of some geometric objects," vol.153, no.4, pp.765–767, 1963.
[21] V. Rvachev, Methods of Logic Algebra in Mathematical Physics, Naukova Dumka, Kiev, 1974. In Russian.
[22] V. Rvachev, Theory of R-functions and Some Applications, Naukova Dumka, Kiev, 1982. In Russian.
[23] A. Biswas and V. Shapiro, "Approximate distance fields with non-vanishing gradients," Graphical Models, vol.66, no.3, pp.133–159, May 2004.
[24] L. Barthe, N.A. Dodgson, M.A. Sabin, B. Wyvill, and V. Gaildrat, "Two-dimensional potential fields for advanced implicit modeling operators," Computer Graphics Forum, vol.22, no.1, pp.23–33, 2003.
[25] J.C. Hart, "Distance to an ellipsoid," in Graphics Gems IV, ed. P. Heckbert, pp.113–119, Academic Press, Boston, 1994.
[26] R. Cartwright, Geometric aspects of empiricla modelling: issues in design and implementation, Ph.D. thesis, University of Warwick, UK, 1998.
[27] A. Pasko, V. Pilyugin, and V. Pokrovskiy, "Geometric modeling in the analysis of trivariate functions," Computers and Graphics, vol.12, no.3/4, pp.457–465, 1988.
[28] W. Lorensen and H. Cline, "Marching cubes: a high presolution 3d surface construction algorithm," Computer Graphics, vol.21, no.4, pp.163–169, 1987. Proceedings of SIGGRAPH 87.
[29] V. Kumar and D. Dutta, "An approach to modeling multi-material objects," pp.336–345, 1997. Fourth Symposium on Solid Modeling and Applications.

Pierre-Alain Fayolle is a PhD candidate in Computer Science at the university of Aizu in Japan. He got his Master in Computer Science from ENSEIRB in France. His research interests include solid and volume modeling.

Benjamin Schmitt is a researcher in Japan at DMProf Company (Digital Media Professional) and at the Computer Graphics Research Institute of Hosei university. He has been studying for his PhD at Bordeaux University, in the LaBRI, Laboratoire Bordelais de Recherche en Informatique. He received his PhD in computer science in 2002 after working on his thesis at Bordeaux University and in collaboration with the Universities of Aizu and Hosei . He is interested in various branches of computer graphics, namely solid modeling, heterogeneous objects, computer art and virtual reality.

Yuichiro Goto was born in 1977. He received the B.S. and M.S. degrees in computer science from the University of Aizu in Japan in 2000 and 2002, respectively. His research areas are geometric modeling, computer graphics, virtual reality, and operating systems. He is a researcher at IT Institute of Kanazawa Institute of Technology.

Alexander Pasko is a professor at Hosei University in Japan. He received M.Sc. and Ph.D. degrees in computer science from Moscow Engineering Physics Institute (MEPI, Russia) in 1983 and 1988. He was a researcher at MEPI from 1983 to 1992 and an assistant professor at the University of Aizu (Japan) from 1993 to 1999. His research interests include solid and volume modeling, animation, multidimensional visualization, multimedia, and computer art. He is a member of ACM SIGGRAPH, IEEE and Eurographics Association.

Constructive heterogeneous object modeling using signed approximate real distance functions

Pierre-Alain Fayolle, Alexander Pasko, Benjamin Schmitt, Nikolay Mirenkov

The University of Aizu, Department of Information Systems
Hosei University
Computer Graphics Research Institute and Hosei University, Digital Media Professional
The University of Aizu, Department of Information Systems

ABSTRACT

We introduce a smooth approximation of the *min* / *max* operations, called SARDF (Signed Approximate Real Distance Function), for maintaining an approximate signed distance function in constructive shape modeling. We apply constructive distance-based shape modeling to design objects with heterogeneous material distribution in the constructive hypervolume model framework. The introduced distance approximation helps intuitively model material distributions parameterized by distances to so-called material features. The smoothness of the material functions, provided here by the smoothness of the defining function for the shape, helps to avoid undesirable singularities in the material distribution, like stress or concentrations. We illustrate application of the SARDF operations by two- and three-dimensional heterogeneous object modeling case studies.

Keywords

Constructive heterogeneous object modeling, distance function approximation, set-theoretic operations, Function Representation (FRep).

1 INTRODUCTION

Solid modeling methods have mostly focused so far on developing models that capture only the geometry of objects, under the assumption that most of them are homogeneous. Recently, a particular attention has been paid to heterogeneous objects modeling, where an object has a number of non-uniformly distributed attributes assigned at each point and varying in space. These attributes may or may not be continuous and have different nature such as photometric characteristics, material density or distribution, physical properties, and others. Heterogeneous objects are widely used in different areas of design and engineering such as rapid prototyping, physical simulations, geological and medical modeling.

We provide in this work new functional definitions for the set-theoretic operations to be used in constructive distance-based modeling of heterogeneous objects. These functions provide a smooth controlled approximation of *min* / *max* used traditionally in constructive modeling of distance fields ([1,2]). In the present work, we try to propose an answer to the question on how can one practically construct heterogeneous objects where the material distributions are parameterized by the distance to material features.

1.1 Previous works

1.1.1 Heterogeneous modeling

Several techniques for modeling heterogeneous objects are already available, presenting some noticeable analogies with homogeneous object modeling. Existing homogeneous object models include surface representations (boundary representation, feature based models); and volumetric representations (voxel arrays, adaptive spatial decompositions, Function Representation, etc). Each homogeneous model has been extended to allow for the underlying model to handle heterogeneity.

R-sets are considered as a basis for modeling and are extended for material inclusion in [3]. An object is subdivided in components; each of them is homogeneous inside and has an assigned material index. Set-theoretic operations can be applied to the solid's components with the corresponding operations on the material. Unfortunately, this modeling technique is limited to the representation of discretely varying material properties. In [4], a more general model is proposed: the geometry is represented by the point set decomposition into a finite set of closed 3-cells, whereas the attributes are defined by a collection of functions, which map the object geometry to several attributes. Such a mathematical model is known as a fiber bundle, with the geometrical model playing the role of the base space. Several other works are using the same model, extending it in various directions ([5,6]). However, as noticed in [7], this model does not really offer concrete computational solutions.

Volumetric representations naturally define solids: a homogeneous object can be defined as a subset of the 3D space, with an additional scalar value given at any point. In the case of a spatial enumeration, like voxels [8], extension to heterogeneity consists in adding a scalar

value for each attribute [9]. The drawback of this method is the difficulty to directly describe the material distribution, without using a data acquisition device (therefore it is supposed that the object to be modeled already exists). Furthermore, the discrete property of the model requires some special approximation procedures.

In [10], a general mathematical framework for modeling heterogeneous objects is proposed. An object is defined as a multidimensional point-set in space; its geometry is characterized by a signed real function at least continuous [11]; different attributes are also defined by functions and their domains of definition. In this framework, the domain of definition of an attribute is called a *space partition*. Both the geometry and the space partitions of the object can be defined by constructive modeling, using either the general R-functions [12], or *min* / *max* functions [1,2]. However, the problem of parameterization and control of attributes in the case of material modeling is left unanswered.

A continuous volumetric representation was proposed in [13], where a B-spline volume is used to model the object geometry, whereas the attributes are modeled by means of diffusion. This model seems to suffer from the lack of flexibility of the geometry limited to volume splines.

Biswas et al. [7] are interested in the representation and control of material distributions by some intuitive parameters related to the geometry of the solid and/or its material features for meshfree modeling. They propose to use the distance functions from material features (point-sets of any dimension with known material properties) as these parameters. It appears from the existing literature that the (Euclidean) distance, or functions of the distance function are indeed the most common types of material functions constructed by methods based on spatial discretization [14,15,16]. The authors of [7] demonstrate that this approach is theoretically complete as it can represent all material functions. However, in their work the modeling of the solid geometry and the material features by Euclidean distance fields is practically not considered.

1.1.2 Signed distance function construction

The Euclidean distance field (or Euclidean distance function) for a given point-set S in the Euclidean space E^n, is a mapping that associates with each point of E^n a real value corresponding to the shortest distance between the given point and every other point in S. Distance fields already have numerous applications in geometric modeling [17], shape metamorphosis [18], object reconstruction from cross-sections [19], robust rendering with sphere tracing [20], generation of skeletal shape representation [21], and other areas. However, it should be noticed that the term distance is often used for different types of distance functions (algebraic, Euclidean, or others) without precise specification. A comparison between Euclidean and algebraic distances to quadric surfaces [22] suggests that the Euclidean one is preferable. In the remainder of the paper, the term distance always refers to the Euclidean distance, unless explicitly specified.

The signed distance function describing a solid is defined as the Euclidean distance from the current point to the surface of the solid, with a sign indicating whether the point is inside or outside the solid [23]. The construction of the signed Euclidean distance function or at least its approximation for the given shape is rather problematic. Deriving an analytical expression is a tedious or sometimes even impossible work. Interpolation methods can be used to provide numerical evaluation at any point of the signed distance function. Given a solid S, the distance to S (its surface) can be sampled on a spatial grid (see [24,25] and references therein) and then interpolated using some basis functions (see [26] and references therein).

Level-set methods [27] can be used to reconstruct an implicit surface (as the zero iso-level of an approximated signed distance function) from unorganized point-sets [28]. However as noticed by Ohtake et al [29] the method is expensive in time and memory. Though the solution is an implicit surface, defined by a signed distance function, its values are known only at the grid nodes and correspond to the solution of a partial derivative equation given by numerical procedures.

Both level-set methods and interpolation may suffer from numerical issues and a loss of accuracy depending on factors such as the choice of the basis, the sampling of the discrete distance field, or the quality of the input data. Finally, both of these methods rely either on: the existence of a discrete distance field (the values of the distance function are known on a finite number of nodes on a grid) or the construction of this discrete distance field from a set of discrete points (and normals) on the surface of the shape (see [24,25] and references therein). As a consequence, both of these methods require that the object already exists (either as a computer model or physically and ready to be scanned), and redo its sampling and the numerical computations if the original solid is modified. Such methods are attractive for already existing objects acquired, for example, by various scanning devices (like laser scanner, MRI, CT, and others). Level-set methods have also an advantage of topological flexibility, useful in structural design and optimization; Wang and Wang [30] use this flexibility to design and optimize heterogeneous objects within a variational framework.

When solids are modeled with a constructive approach which is the case in some engineering fields, it would appear logical to support building the distance function in a constructive way as well.

Constructive modeling is based on applying successively set-theoretic or other operations to predefined shapes (primitives). Let two solids (point-sets in Euclidean space) be denoted by $f_1 \geq 0$ and $f_2 \geq 0$, then the union of these two objects can be defined by $f_\cup = max(f_1, f_2)$ and the intersection by $f_\cap = min(f_1, f_2)$ [1,2]. In this case, if $f_1(x, y, z)$ and $f_2(x, y, z)$ correspond to the signed distances from the point (x, y, z) to the surfaces defined by $f_1 = 0$ and $f_2 = 0$, then $f_\cup$ and $f_\cap$ correspond to approximate

distance functions for the entire complex object (see [20] for a proof that the resulting function defining the complex object is not the exact distance function but is bounded by the exact distance function). However, *min* / *max* operations result in "C^1 discontinuity" at any point where $f_1 = f_2$. It can cause unexpected results in further operations on the object such as blending, metamorphosis, and others.

There are works aiming at the replacement of *min* / *max* with smoother functions to overcome this "C^1 discontinuity": Rvachev proposed the R-functions [12], which define exact set-theoretic operations, but do not preserve the Euclidean distance value. R-functions also suffer from an exponential growth of the function value when for example the union of several overlapping solids is considered. A normalization can be applied but it approximates the distance function only close to the surface. The constructive approach of [31] relies on joining trimmed lines (for curve modeling) or trimmed triangles (for surface modeling) by R-functions. The result is then normalized to avoid bulging on the joint points. This approach can smooth the resulting distance function to any order, but at the expense of the quality of the distance approximation, which is accurate only close to the surface. This method may also involve a significant number of segments, resulting in a potentially large size for the representation. Moreover the problem of deriving a constructive representation from trimmed implicit surfaces or curves is related to the problem of boundary to CSG (Constructive Solid Geometry) conversion [32,33] and is not fully solved. The superelliptic approximations of *min* / *max* [1] do not describe exact set-theoretic operations and suit only to blending. The elliptic approximation of *min* / *max* by Barthe et al [34] is designed initially for blending and the approximation error grows infinitely far from the boundary.

1.2 Problem statement and contribution

We consider the problem of constructive modeling of heterogeneous objects, where the distance to the boundary surface of the solid and/or to the material features is used as a parameter to control the material functions. The work of [7] is extended by the proposed controlled smooth approximation of the Euclidean distance field for designing the geometry of the solid and its material attributes in a constructive way.

For this purpose, we introduce (section 2) new approximations for *min* / *max* operations inspired by the work of [34]. The new proposed functions are in C^1 on $R^2\backslash\{(0,0)\}$ and keep a bounded and controllable approximation of the *min* / *max* functions. From this point of view, we call the constructed defining function of the object by the term signed approximate real distance function (SARDF), the approximate *min* function can be called SARDF intersection, and the approximate *max* function - SARDF union.

These set-theoretic operations are used within the hypervolume constructive framework of Pasko et al. [10] and extend it for distance based modeling. In the latter model, both the geometry of the solid, and the shape of the definition domains for the attributes can be defined in constructive ways. Under the condition that primitives are defined by Euclidean distance functions (or approximation), the resulting constructive solids and attributes geometry built with the proposed operations have a smoothed distance like field as their defining function (section 3). Euclidean distance functions are available for all typical primitives of a traditional CSG system [20]. Normalized primitives (Appendix A of [31]) can also be used though the resulting distance functions will approximate the distance only close to the surface. Interpolation methods can also be used to define some complex freeform shapes as primitives.

The modified constructive hypervolume model is used to answer the question (section 5.2 of [7]) of the practical ways of computing the Euclidean distance field and then is combined with the work of [7] to model constructive heterogeneous objects with signed distance fields. Section 4 presents some examples of heterogeneous objects constructed with the proposed approach. We also provide a comparison with other existing set-theoretic operations and underline the quality and accuracy of the approximated Euclidean field (compared to R-Functions), and its smoothness (compared to *min* / *max*).

2 SARDF FRAMEWORK

Our intention is to propose a framework for modeling constructive heterogeneous objects where the Euclidean distance to the object and material features' boundary is used as a parameter to control the material distributions. The idea of parameterization of the material distribution by distance is already discussed in several works (see [7] and the various references therein), whereas the construction of the Euclidean distance function is practically not discussed.

We rely on the mathematical model introduced by Pasko et al. [10] to define heterogeneous objects. This model is reminded in the following sub-section. It is extended by primitives defined with signed Euclidean distance functions, and new formulations for the set-theoretic operations, which keep a better distance approximation than the R-functions and are smoother than *min* / *max*.

2.1 Constructive hypervolume model

The constructive hypervolume model is introduced in [10] as a mathematical model to define heterogeneous objects. A general hypervolume object is defined as a multidimensional point set G with multiple attributes given at any of its points. The attributes S_i represent abstract values or physical characteristics such as temperature, color, material distribution, etc. A representation of the hypervolume is proposed as:

$o = (G, A_1, \ldots, A_k) : (F(X), S_1(X), \ldots, S_k(X))$ where:

- $X = (x_1, \ldots, x_n)$ is a point in the n-dimensional Euclidean space E^n,

- $F : E^n \rightarrow R$ is a real-valued function of point coordinates to represent the point set *G*, based on the FRep model [11],
- $S_i : SP_i \rightarrow R$, $SP_i \subset E^n$ is a real-valued scalar function corresponding to an attribute A_i that is not necessarily continuous.

The function *F(X)* is real valued. For each given point, it is evaluated and depending on the sign of the returned value, one can classify the given point as inside, outside or on the boundary of the object. This function is represented in the modeling system by a tree structure with primitives in the leaves and operations in the nodes. The term *constructive tree* is generally used for this tree structure. The only requirement of the FRep model is that the defining function *F* is at least continuous.

Similarly, depending on the applications, the attribute functions can be defined using physical models or a constructive approach. The spatial subset, where an attribute is defined, is called a *space partition,* designated as SP_i in the above formulation. There are no definite values for an attribute outside its space partition. For each material feature, there is at least one space partition, containing this material feature. However a material feature can be contained in more than one space partition, in the case, for example, when the material feature is made of the known composition of several materials.

In the present work, the former model is extended so that each primitive is defined by the Euclidean distance function. Hart [20] provides a list of primitives with known distance functions. These primitives are used to define both the geometry of the solid and the space partitions (i.e. the spatial subset where an attribute is defined). Set-theoretic operations on primitives and complex objects are implemented with the proposed SARDF operations.

2.2 Overview of SARDF operations

Any contour line of the *min* and *max* functions has a sharp corner at the point, where the two arguments have equal value. Following the general approach of [34], we propose to replace the sharp corner in any contour line with a circular arc. Two parabolas, symmetric with respect to the line $y = x$, are used to delimit the frontier of the circular arc approximation (see Fig. 1 in the case of the smoothed *min* function).

The use of circular approximations for the *min* and *max* functions provides the smoothness property of the resulting approximate distance function for constructive shapes built using normal primitives (i.e. defined by Euclidean distance functions). We prevent the radius of the circular arc from growing infinitely by introducing a fixed threshold *R*. A bounding band is introduced by two parallel straight lines that enclose the arcs with this fixed radius. These band lines are defined by a shift of the line $y = x$ at *R* distance in positive and negative *x* directions: $y = x - R$ and $y = x + R$ (see Fig. 1). This bounding band provides a fixed upper limit of the error compared to the *min* / *max* approach at any given point.

The two branches of the parabolas are defined to be tangent to the two parallel lines $y = x - R$ and $y = x + R$ at the connecting points *(R, 2R)* and *(2R, R)* and pass through the origin *(0,0)*; it gives the expressions for these two parabolas: $y = \frac{x^2}{4R}$ and $x = \frac{y^2}{4R}$. Note that the use of parabolas to restrict the circular approximation ensures that the constructed function is C^1 on the arc of circle A_1A_2.

The resulting defining functions of shapes are built using normal primitives and the newly introduced set-theoretic operations called SARDF operations. The union operation is called SARDF union (we use the symbol "$\cup_S$") and the intersection operations is called SARDF intersection (with the symbol "$\cap_S$").

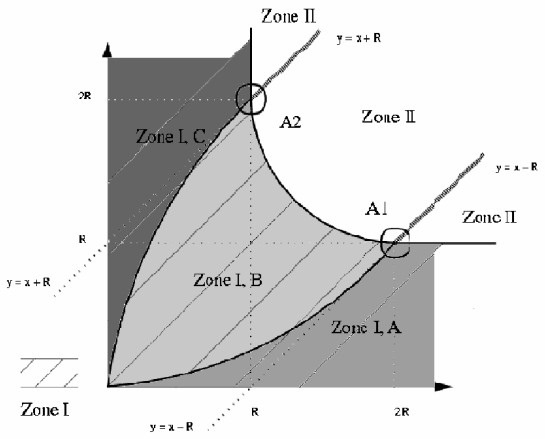


Figure 1: The first quadrant is divided into two zones. The growing circular approximation is applied in zone I, whereas we introduce a fixed radius approximation with the bounding band in zone II.

2.3 SARDF intersection of arbitrary objects

2.3.1 Construction of the intersection function

The main steps of the construction of the smoothed intersection in the first quadrant ($x > 0$ and $y > 0$) are given. By symmetry, the construction in the third quadrant ($x < 0$ and $y < 0$) is similar. In the two other quadrants, the expression for the intersection is equal to the *min* function.

Zone I, B Given a point *(x, y)* in the first quadrant we consider the first case when it is in zone I, B in Fig. 1 and calculate the iso-level value *d* for the smoothed intersection function at this point. The given point belongs to a circular arc that is tangentially connected to two horizontal and vertical rays when reaching the parabola (see Fig. 1). The equation of this arc is $(x - x_0)^2 + (y - y_0)^2 = r^2$, where x_0, y_0 and *r* need to be expressed as functions of the searched value *d*. The point at the intersection of the parabola and the iso-level *d* of the

searched function, is at a distance d from the axis $y = 0$. This point belongs also to the parabola, so it satisfies $d = \frac{x_0^2}{4R}$, and by symmetry: $d = \frac{y_0^2}{4R}$.
The coordinates of the center of the circular arc (x_0, y_0) satisfy: $x_0 = y_0 = d + r$; it follows that $r = y_0 - d = 2\sqrt{Rd} - d$. Using the substitution of variables $\sqrt{d} = z$ and expanding the equation of the circular arc, we obtain the following algebraic equation:

$$z^4 - 4\sqrt{R}z^3 - 4Rz^2 + 4\sqrt{R}(x+y)z - (x^2+y^2) = 0 \quad (1)$$

Thus in the first quadrant, in the zone I, B, the expression of the intersection is the square of one of the four roots of the algebraic equation (1). The roots of an algebraic equation of degree four are known algebraically. The root of interest is found by using one of the limit conditions, for example $d(2R,R) = R$.

Zone II, inside the bounding band Given a point (x, y) in zone II, within the bounding band (see Fig. 1), the iso-level value d of the smoothed intersection function at that point is searched. This point belongs to a circular arc that is tangentially connected to two horizontal and vertical rays when reaching the two lines of the bounding band. The equation of this circular arc is: $(x - x_0)^2 + (y - y_0)^2 = R^2$. Note that R is a constant, only x_0 and y_0 need to be expressed as functions of d.
The coordinates (x_0, y_0) of the circular arc satisfy: $x_0 = y_0 = d + R$. After substitution into the equation of the circular arc and expanding this equation, d is obtained as one of the two solutions of the following algebraic equation:

$$2d^2 + d(4R - 2x - 2y) + (x^2 + y^2 - 2R(x+y) + R^2) = 0 \quad (2)$$

The root of interest is obtained by using the limit condition: $d(2R, R) = R$.

Zone I, A and C and II outside the bounding band The expression of the intersection is exactly equal to *min* in these areas.

2.3.2 Expression for the intersection operation

Let $f_1(x,y,z)$ and $f_2(x,y,z)$ be the distance functions defining two objects. The intersection between these two objects is defined following the previous method with the substitution $x \rightarrow f_1(x,y,z)$ and $y \rightarrow f_2(x,y,z)$. The general expression for the SARDF intersection function of f_1 and f_2 can be described as follows:

Case 1: $f_1 > 0$ **and** $f_2 > 0$ In the current paragraph, E_1 is used for the following boolean expression: $E_1 = (f_1 < R$ or $f_2 < R$ or $(f_1 < 2R$ and $f_2 < 2R$ and $(f_1 - 2R)^2 + (f_2 - 2R)^2 > R^2))$.

- If E_1 and $f_2 > \frac{f_1^2}{4R}$ and $f_1 > \frac{f_2^2}{4R}$, $f_1 \cap_S f_2 = z^2$, where z is the root of (1) verifying $z^2(2R,R)=R$;
- if E_1 and $f_2 \le \frac{f_1^2}{4R}$, $f_1 \cap_S f_2 = f_2$
- if E_1 and $f_1 \le \frac{f_2^2}{4R}$, $f_1 \cap_S f_2 = f_1$
- if $\neg E_1$ and f_1-R < f_2 < f_1 + R, $f_1 \cap_S f_2 = \frac{1}{2a}(-b + \sqrt{b^2 - 4ac})$, where $a = 2$, $b = -2f_1 - 2f_2 + 4R$, and $c = f_1^2 + f_2^2 - 2f_1R - 2f_2R + R^2$;
- if $\neg E_1$ and $f_2 \le f_1$-R, $f_1 \cap_S f_2 = f_2$
- if $\neg E_1$ and $f_2 \ge f_1$+R, $f_1 \cap_S f_2 = f_1$

Case 2: $f_1 \le 0$ **and** $f_2 \ge 0$
$f_1 \cap_S f_2 = f_1$

Case 3: $f_1 < 0$ **and** $f_2 < 0$ We denote by E_2 the following boolean expression: $E_2 = (f_1 > -R$ or $f_2 > -R$ or $(f_1 > -2R$ and $f_2 > -2R$ and $(f_1 + R)^2 + (f_2 + R)^2 < R^2))$.

- if E_2 and $f_2 < -\frac{f_1^2}{4R}$ and $f_1 < -\frac{f_2^2}{4R}$, $f_1 \cap_S f_2 = z$, where z is the root of $\frac{d^4}{16R^2} - \frac{d^3}{2R} + \frac{1}{2R}d^2(f_1 + f_2 - 2R) + (f_1^2 + f_2^2) = 0$ verifying z(-2R,-R)=-2R;
- if E_2 and $f_2 \ge -\frac{f_1^2}{4R}$, $f_1 \cap_S f_2 = f_2$
- if E_2 and $f_1 \ge -\frac{f_2^2}{4R}$, $f_1 \cap_S f_2 = f_1$
- if $\neg E_2$ and f_1-R < f_2 < f_1 + R, $f_1 \cap_S f_2 = \frac{1}{2a}(-b + \sqrt{b^2 - 4ac})$, where $a = 2$, $b = -2f_1 - 2f_2 + 4R$, and $c = f_1^2 + f_2^2 - 2f_1R - 2f_2R + R^2$;
- if $\neg E_2$ and $f_2 \le f_1$-R, $f_1 \cap_S f_2 = f_2$
- if $\neg E_2$ and $f_2 \ge f_1$+R, $f_1 \cap_S f_2 = f_1$

Case 4: $f_1 \ge 0$ **and** $f_2 \le 0$

$f_1 \cap_S f_2 = f_2$

2.4 Smoothness of the SARDF intersection

We give the main points of the proof that the function SARDF intersection is in C^1 on $R^2 \setminus \{(0,0)\}$. At $(0,0)$ the function is continuous only.
In the quadrants II and IV the function is C^1. The proof is similar in the quadrants I and III, so it is given only in the first quadrant.
First, we note that in each zone of the first quadrant, the different parts of the function are C^1. The problems of continuity of the function and the partial derivatives may appear at the boundaries between the different

expressions, i.e., on the branches of the parabolas, the straight lines ($y = x - R$ and $y = x + R$), and on the circular arc boundary A_1A_2 between the growing radius zone and the fixed radius zone. The function is symmetric with respect to the line $y = x$, thus only the part of the first quadrant below $y = x$ needs to be considered.
Note that for two general objects defined by $f_1(x,y,z)$ and $f_2(x,y,z)$, the partial derivatives are obtained with the substitution $x \rightarrow f_1(x,y,z)$ and $y \rightarrow f_2(x,y,z)$, and the Jacobian of $(f_1(x,y,z), f_2(x,y,z))$. The smoothness of the constructed object is dependent on the smoothness of f_1 and f_2.

2.4.1 SARDF intersection is continuous

At the parabolic boundary, Let $P = (x, y) = (u, \frac{u^2}{4R})$, with $u \in [0, 2R]$ be a point on the parabola arc, the value of the function in zone I, A, at that point is $y = \frac{u^2}{4R}$. We need to check that this value at P matches with the value of the function in Zone I, B at that point, given by equation (1). It is easy to check that for $(x, y) = (u, \frac{u^2}{4R})$, and with $z = \sqrt{d} = \frac{u}{2\sqrt{R}}$, the algebraic relation (1) holds. We can conclude with the continuity of the SARDF expression at the parabolic boundary.

At the line $y = x - R$, Let $P = (x, y) = (u, u-R)$, with $u \in [2R, \infty[$, be a point on the straight line, the value of the function in zone II, outside the bounding band at P is $y = u - R$. Again we check that this value at P matches with the value of the expression within the boundary band, zone II, given by equation (2). It is easy to check that for $P = (x, y) = (u, u-R)$, $d = u - R$ satisfies equation (2), we can conclude with the continuity of the SARDF expression at this boundary.

At the circular arc boundary A_1A_2, let $P=(x, y)=(2R+R\cos(u),2R+R\sin(u))$, with $u \in \left[\frac{5\pi}{4}, \frac{3\pi}{2}\right]$, be a point on that circular arc. The value of the function at P is given by its value at the point A_1 and is $y = R$. We check that this value matches the value of the expressions in the zones I, B and II at P, by checking that equations (1) and (2) hold. Again, after applying some calculus, the relations (1) and (2) hold and we can conclude with the continuity at that boundary.

2.4.2 SARDF intersection is C^1

To prove it, we verify that the value of the partial derivatives match on the boundary points. First, the equation (1) and (2) are used to obtain expressions for the partial derivatives of the function in the zones I, B and II, within the boundary.

Expression for the partial derivatives in zone I, B: In zone I, B, the square root of the function SARDF intersection $\sqrt{d}$ satisfies the algebraic equation (1). Taking the partial derivative with x of (1) gives:

$$4\frac{\partial z}{\partial x}z^3 - 12\sqrt{R}\frac{\partial z}{\partial x}z^2 - 8R\frac{\partial z}{\partial x}z + 4\sqrt{R}(x+y)\frac{\partial z}{\partial x} + 4\sqrt{R}z - 2x = 0$$

It follows that: $\frac{\partial z}{\partial x} = \frac{2x - 4\sqrt{R}z}{4z^3 - 12\sqrt{R}z^2 - 8Rz + 4\sqrt{R}(x+y)}$.
Since $z = \sqrt{d}$, it comes that $\frac{\partial z}{\partial x} = \frac{1}{2}\frac{\partial d}{\partial x}\frac{1}{\sqrt{d}}$. Combining it with the previous expression, we get the following relation for the partial derivative of the function in zone I, B

$$\frac{\partial d}{\partial x} = 2\sqrt{d}\,\frac{2x - 4\sqrt{R}z}{4z^3 - 12\sqrt{R}z^2 - 8Rz + 4\sqrt{R}(x+y)} \quad (3)$$

with $z(x,y) = \sqrt{d(x,y)}$.
By the same procedure, we obtain an expression for the partial derivative by y in zone I,B:

$$\frac{\partial d}{\partial y} = 2\sqrt{d}\,\frac{2y - 4\sqrt{R}z}{4z^3 - 12\sqrt{R}z^2 - 8Rz + 4\sqrt{R}(x+y)} \quad (4)$$

with $z(x,y) = \sqrt{d(x,y)}$.

Expression for the partial derivatives in zone II, within the bounding band: In zone II, within the bounding band, the function satisfies equation (2). With the same method as above, we take the partial derivative by x gives: $4\frac{\partial d}{\partial x}d + \frac{\partial d}{\partial x}(4R - 2x - 2y) - 2d + (2x - 2R) = 0$.
It follows that:

$$\frac{\partial d}{\partial x} = \frac{2d - 2(x - R)}{4d + (4R - 2x - 2y)} \quad (5)$$

Similarly an expression for the partial derivative by y can be obtained:

$$\frac{\partial d}{\partial y} = \frac{2d - 2(y - R)}{4d + (4R - 2x - 2y)} \quad (6)$$

Expression for the partial derivatives in zone I, A and zone II, below $y = x-R$**:** The function is exactly *min* in these two areas, and so the partial derivative in the x direction is 0 and 1 in the y direction.

"C^1 continuity" at the parabolic arc: Let $P = (x, y)$

$=(u, \frac{u^2}{4R})$, with $u \in [0, 2R]$, be a point on the arc, at P, $z = \sqrt{d} = \frac{u}{2\sqrt{R}}$; it is easy to verify that at P equations (3) and (4) give: $\frac{\partial d}{\partial x} = 0$ and $\frac{\partial d}{\partial y} = 1$.

"C^1 continuity" at the line $y = x - R$: Let $P = (x, y) = (u, u-R)$, with $u \in [2R, \infty[$ be a point on the line, at P, $d = u - R$; it is easy to verify that at P equations (5) and (6) give: $\frac{\partial d}{\partial x} = 0$ and $\frac{\partial d}{\partial y} = 1$.

"C^1 continuity" at the circular arc boundary: Let $P=(x, y)=(2R+R\cos(u), 2R+R\sin(u))$, $u \in \left[\frac{5\pi}{4}, \frac{3\pi}{2}\right]$ be a point on the circle boundary between the growing radius zone and the constant radius zone. At P, the value of the function is R. Using these informations in equation (3), we obtain after straightforward calculus $\frac{\partial d}{\partial x} = \frac{\cos(u)}{\cos(u) + \sin(u)}$. Similarly, with equation (5): $\frac{\partial d}{\partial x} = \frac{\cos(u)}{\cos(u) + \sin(u)}$.

Similarly for the partial derivative $\frac{\partial d}{\partial y}$, equation (4) gives: $\frac{\partial d}{\partial y} = \frac{\sin(u)}{\cos(u) + \sin(u)}$, and equation (6) gives: $\frac{\partial d}{\partial y} = \frac{\sin(u)}{\cos(u) + \sin(u)}$.

With the equality of the partial derivatives we can conclude that the SARDF intersection is C^1 on the boundary circular arc. It is still C^1 at A_1, with $u = \frac{3\pi}{2}$.

3 COMPARISON OF SARDF WITH OTHER SET-THEORETIC OPERATIONS

3.1 Time comparison

We look at the time efficiency of SARDF union and intersection and the overhead in time compared to *min* / *max* and the R-functions union and intersection.

According to the tests, we found a factor of approximately 6 between SARDF union and *max*, 5.5 between SARDF intersection and *min*, 3.5 between SARDF union and R-function union and 2.9 between SARDF intersection and R-function intersection. The sampling of the SARDF intersection on a 20001*20001 grid takes 29.5 seconds (5.537 for max, and 10.074 for the R-function). This sampling corresponds to sampling a model with 400 intersection operations on a 100*100*100 grid. All the functions in the tests were implemented in C language on a Pentium 4 processor 1.7 GHz, with 256 MBytes of RAM. No particular optimization was used in the code, learning room for improvement.

3.2 Qualitative comparison of the union operations: SARDF union, max, and R-union

We qualitatively compare the three union operations *max*, R-union, and SARDF union applied in constructive modeling in terms of the quality of approximation of the Euclidean distance for the resulting function, and in terms of the smoothness of this function. The solid used in the experiments is the union of two ellipsoids centered at (5, 0, 0) and (5, 0, -5) with respective radii (5, 2, 2) and (2, 2, 5).

Let f_1 and f_2 be the signed Euclidean distance functions defining the two ellipsoids. The interior of an ellipsoid corresponds to $f_i > 0$ and the exterior to $f_i < 0$. The surface boundary of an ellipsoid corresponds to $\{X \in R^3: f_i(X)=0\}$. The procedure used for computing the exact signed distance from a point in R^3 to the surface of an ellipsoid is discussed in [35].

We are interested in the quality of the approximation of the Euclidean distance function and the smoothness of $f_1 \cup f_2$, where $\cup$ stands for the functional definition of one of the three union operations. The expression used for the R-union of two objects is $f_1 \cup f_2 = f_1 + f_2 + \sqrt{f_1^2 + f_2^2}$.

Figure 2 shows the contour maps of a cross-section by the plane $y = 0$ of the resulting defining functions for different analytical expressions for union.

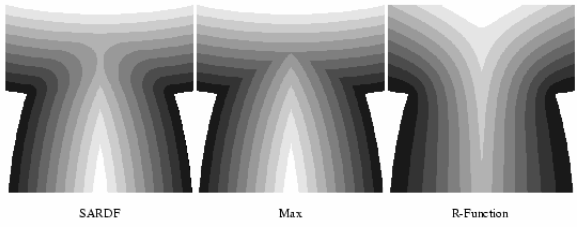


Figure 2. Contour maps of the union of two ellipsoids for the three different union operations. The level of gray corresponds to range of approximate distances from points inside the solid to its surface. Left: SARDF union, middle: *max*: sharp corners indicate points of derivatives discontinuity, right: R-union: contour map of the function clearly shows that the resulting function loses the distance like behaviour quite close to the boundary even with exact distance functions used for the arguments.

Figures 2 left, middle and right show some interior contour lines of the function for the considered object. The figures correspond to the models made respectively with SARDF union, *max*, and R-function union. The contour map of the defining function built with R-union loses the Euclidean distance like behavior quite close to the surface boundary even with exact distance functions defining the two ellipsoids. This loss of the distance like

property is emphasized with the comparison to the maps given by using SARDF union and *max* (Fig. 2 left and middle). These contour maps have very similar contour lines except at the points joining the contour lines of the two functions, which are sharp for *max* (see Fig. 2 middle) and smooth for SARDF (see Fig. 2 left). It illustrates the discontinuity of the partial derivatives of the function defining the solid and built using *max*.
Points of discontinuity of the Euclidean distance's partial derivatives are also present on the main axis of each ellipsoid (Fig. 2). It is known that as soon as at least two points of the shape have equal distance values to the given point in space, the Euclidean distance function has a discontinuity of its partial derivatives at this point. This set of points is known as the medial axis of a shape. The discontinuity of the partial derivatives at these points remains independantly of the analytical expression used for the union operation.
So far, we have illustrated two properties: the SARDF operations do not introduce additional points where the resulting function is not C^1, like *min* / *max*; and they are better approximations of the Euclidean distance than the R-functions. Both these properties are important in heterogeneous object modeling: Shin and Dutta relate that R-function is not an exact distance function, consequently making it difficult for a designer to predict or control the material distribution (section 3.3 p. 210 and Figure 10, p. 211 of [36]). Biswas et al report that the lack of smoothness in a material function parameterized by the distance result in zones of concentration or stress of the materials (section 1.3 of [7]).

4 CONSTRUCTIVE HETEROGENEOUS OBJECTS MODELING WITH SARDF

We propose some examples to illustrate the use of the SARDF operations and normal primitives in constructive heterogeneous modeling. The SARDF operations are used instead of the R-functions or the *min* / *max* functions in the different constructive trees to define the geometry of the solid and the space partitions where attributes are defined.
A normal primitive, is a primitive with a defining function *f*, which at a given point $X \in R^3$, returns the Euclidean distance from X to the surface $f^{-1}(0)$. Primitives with known expressions for the distance are given in [20].
We show how the different expressions for the set-theoretic operations affect the material distributions and their properties.

4.1 Two-dimensional example

At first, we illustrate the use of SARDF in modeling a two-dimensional heterogeneous object. The geometry of the object (Fig. 3) is defined as $f(X) \geq 0$, where *f* is evaluated by traversing the constructive FRep tree [11] with a box and a cylinder in the leaves, and the subtraction operation in the node.

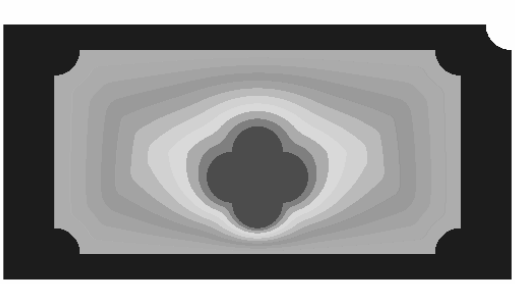


Figure 3. A two-dimensional CAD part with three different material regions (blue: material 1, red: material 2, color gradient: functionally graded material.

This object is made of two materials and three material regions (Fig. 3). We use the notation $m_1(X)$ and $m_2(X)$ for the scalar volume fraction component of the materials 1 and 2. For visualization purposes, the material distributions are mapped to the "RGB" color space: a color is attributed to each material and the final color is the combination of the colors corresponding to each material, weighted by the scalar volume fraction.
Two of the three material regions correspond to regions where there is only material 1 uniformly distributed (blue in Fig. 3) and there is only material 2 uniformly distributed (red in Fig. 3) correspondingly. The last material region corresponds to functionally graded material. The geometry of each region is defined using FRep in a constructive way, similarly to the shape's geometry. SARDF operations are used in the nodes and normal primitives in the leaves of the constructive tree. The resulting functions provide C^1 approximation of the distance to each material region. These distances are used to specify the functionally graded material.
The scalar volume fraction of each component material in the functionally graded material region is given by: $m_1(X)=w_1(X)M_1$ and $m_2(X)=w_2(X)M_2$, where M_1 and M_2 stand for the value of the scalar volume fraction on the boundary of the first and second material features shown respectively in blue and red Fig. 3.
The weighting functions $w_1(X)$ and $w_2(X)$ are defined using a normalization of each inverse distance functions:

$$w_1(X) = \frac{\frac{1}{d_1(X)}}{\frac{1}{d_1(X)} + \frac{1}{d_2(X)}} = \frac{d_2(X)}{d_1(X) + d_2(X)} \quad (7)$$

$$w_2(X) = \frac{\frac{1}{d_2(X)}}{\frac{1}{d_1(X)} + \frac{1}{d_2(X)}} = \frac{d_1(X)}{d_1(X) + d_2(X)} \quad (8)$$

where $d_1(X)$ and $d_2(X)$ are the distances from point X to the boundary of respectively the material features shown in blue and red Fig. 3.
These two distance maps are illustrated in Fig. 4 and Fig. 5. Fig. 4, left and Fig. 4, right correspond respectively to the approximate distance map d_1 when the R-functions and the SARDF operations are used correspondingly to define the shape. In a similar way, Fig. 5, left and Fig. 5, right correspond to the approximate distance d_2 when using R-functions and SARDF.
The approximate distance maps built using R-functions

indicate that even if R-functions have good smoothness properties, they do not provide a good approximation to the distance function, making it difficult to precisely control the material distribution.

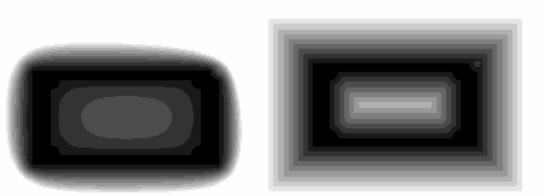


Figure 4. Approximate distance map d_1 from point X to the boundary of the region where only material 1 exists. Left: using R-functions. Right: using SARDF operations.

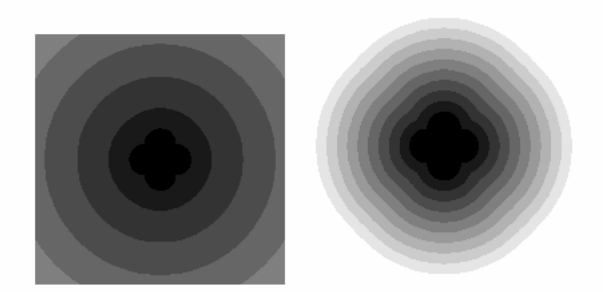


Figure 5. Approximate distance map d_2 from point X to the boundary of the region where only material 2 exists. Left: using R-functions. Right: using SARDF operations.

The weighting functions $w_1(X)$ and $w_2(X)$ are continuous and satisfy the interpolation condition $w_i(\partial B_j) = \delta_{ij}$, with $1 \le i,j \le 2$, δ_{ij} is the Kronecker symbol, and ∂B_j are the boundaries of the material features seen in blue and red Fig. 4. The functions $w_1(X)$ and $w_2(X)$ form a partition of unity.

The properties of these functions are illustrated in Fig. 6 with a cross section of the model through the y-axis and the visualization of the evolution of the weighting functions $w_1(X'=(X,const))$ and $w_2(X')$ along the x-axis. Note that in the current example m_1 and m_2 have the same graphs, since the values of the volume fraction on the boundaries, M_1 and M_2 have been chosen equal to 1.

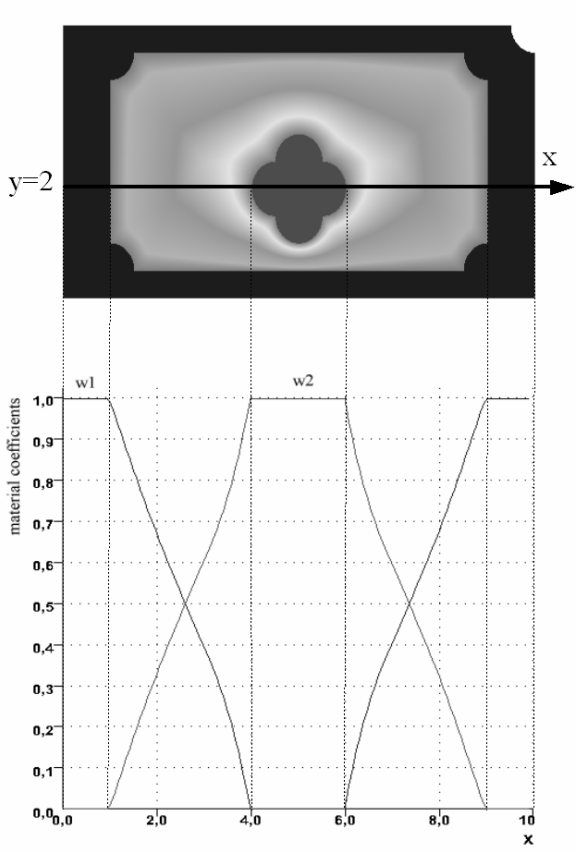


Figure 6. A cross section parallel to x-axis and the distribution of the materials in the cross section for the CAD part constructed with SARDF functions.

One can notice (Fig. 6) a "C^1 discontinuity" at the points on the boundary of the material features. This can cause the same problems as the distance function "C^1 discontinuity". Fortunately, these sharp corners can be smoothened by a modification of the expressions for the coefficients (Eq. (7) and (8)). The expressions used for the material feature weights correspond to a particular case of the inverse distance weighting [37]. More general expressions are:

$$w_1(X) = \frac{d_2^k(X)}{d_1^k(X) + d_2^k(X)} \quad \text{and} \quad w_2(X) = \frac{d_1^k(X)}{d_1^k(X) + d_2^k(X)} \quad .$$

The case $k=1$ gives Eq. (7) and (8). The parameter k controls the smoothness of the functions on the points of the material features.

Replacing every SARDF operation by an R-function or *min* / *max* in the constructive trees for the geometry of the solid and the material regions gives different material distributions in the same cross-section (see Fig. 7, left and 7 right).

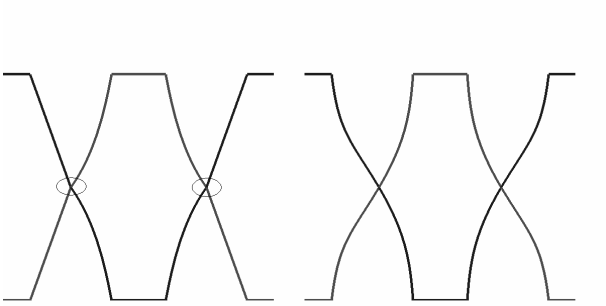


Figure 7. Material distributions in the cross-section $y = 2$ for materials 1 and 2 using: R-functions in the constructive trees for the geometry of the solid and the material regions (right). *min* / *max* in the constructive trees for the geometry of the solid and the material regions (left). The circled points correspond to points of C^1 discontinuity of the material distributions.

Figures 7, left and right reflect at the level of the material distribution the problems of using the R-functions, or *min* / *max* in constructive heterogeneous modeling. Figure 7, right shows the role played by the accuracy of the distance approximation when the distance is used to parameterize the material distributions. The unpredictable behaviour of the distance approximation makes the task of the designer difficult. For example, we would expect that the first part of the blue curve (just before the intersection with the red curve) is linear. This "bad behaviour" of the R-functions was noticed by Shin and Dutta in [36].

Figure 7, left illustrates the "C^1 discontinuity" of the *min* (and *max*) functions and its impact on the material distribution. Both distributions of material 1 (blue) and 2 (red) have two points of "C^1 discontinuity" (circled in Fig. 7, left). It results in problems of stress or concentrations as noticed by Biswas et al in [7].

Using SARDF for the set-theoretic operations does not introduce new points of C^1 discontinuity, and keeps a good approximation of the distance; these properties can be seen consequently in the graph of the material distributions (Fig. 6).

In this and in the following examples only two materials are in the overlapping zone. More materials can be blended and the expressions for inverse distance weighting (Eq. (7) and (8)) can be extended to the case where more than 2 materials are blended. Additional details on the inverse distance weighting used for the interpolation of materials defined over functionally defined sets can be found in [38]. More complex expressions for compositions of multiple materials, like vector valued materials, constrained and weighted interpolation of materials can be found in [7].

The resulting model of the CAD part and its material distribution is illustrated by Fig. 3. The distribution of the material 1, given by its scalar volume fraction $m_1(X)$ is mapped to the blue color, and the distribution of the material 2, given by $m_2(X)$ to the red color. Stripes are used to make the visualization of the changes in material distribution easier.

4.2 Three-dimensional CAD part

We propose a second example (in three-dimensional space) with more complex shapes for the geometry of the object and the geometry of the regions corresponding to the material features. With this example, we underline the fact that complex shapes can be made using SARDF operations and normal primitives.

The overall geometry of the object is a block with two (constant) material features inside. We keep the same notation as in the previous subsection, with $m_1(X)$ and $m_2(X)$ the scalar volume fraction of the materials 1 and 2. Figure 8, top, left shows the first material feature corresponding to the material 1 (in blue); it is cut by a planar half-space for visualization purposes only. Figure 8, top, middle shows the second material feature (in red); its geometry is composed of blocks and ellipsoids, combined with SARDF unions and intersections. Fig. 8, top, right, illustrates a zoom to one of the pins. Such a pin is modeled with ellipsoids as primitives and SARDF union and intersection as operations: it is the SARDF union of four ellipsoids, which are after subtracted from a fifth ellipsoid.

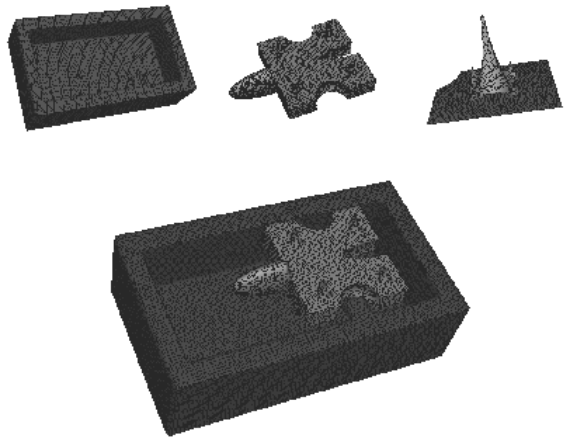


Figure 8. Top, left: the first material feature, top, middle: the second material feature, with a zoom on one of the pins, on the right (top, right), bottom: union of the two material features.

To express the material behaviour in the region between the two material features (this region can be seen in Fig. 10, bottom), we use the equations (7) and (8) for the weights for each material feature. It indicates that the closest material feature has the strongest influence. The overall distribution of the materials is shown in Fig. 9, left. The geometry corresponding to the second material feature is rendered, using a red color, then for the visualization of the material distribution, two cross-sections are made: one for $x=0$ and one for $z=0$. For each of the cross-section, the evolution of the material distribution is projected. For visualization purposes, each material is mapped to one color. The first material corresponds to the blue color and the second material to the red.

Figure 9, right shows a zoom to one of the pins. The geometry of the second material feature is drawn in 3D with red color. Two more cross sections with distribution of materials are added. The gradient of color expresses

the evolution of the distribution of the material composition, indicating percentage of the first and second material.

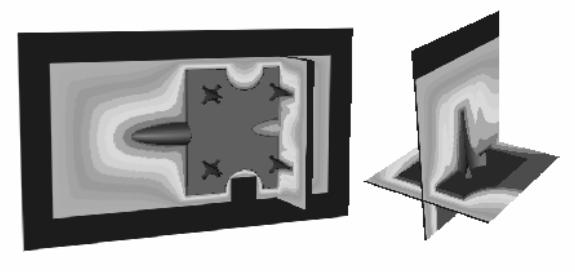


Figure 9. Distribution of two materials. Blue color corresponds to material 1, red color to material 2. The color variation indicates the fraction of each material. Left: Two cross sections are made for $x=0$ and $y=0$ to show the material distribution. Right: A zoom is made to one of the pins with two additional cross-sections.

5 CONCLUSION

5.1 Advantages of the SARDF framework in constructive heterogeneous object modeling

The core of this work is the introduction and application of special functions describing set-theoretic operations for constructive modeling. Under the condition that the primitives in use are defined by exact distance functions, the proposed set of functions provides a good approximation of the real distance value. These functions serve for defining both the geometry of the object and the geometry of regions where the materials are distributed.

The result of any constructive modeling, involving signed approximate real distance functions (SARDF) and primitives defined by distance functions, is at least C^1, except for the cases when both of the arguments of the SARDF operation are equal to 0 (surface-surface intersection curves), or for the points belonging to the medial axis of one of the primitives.

Compared to the known R-functions, the proposed SARDF operations provide better approximations for the Euclidean distance. In contrary to *min* / *max*, they do not introduce extra points of "C^1 discontinuity". Therefore they seem extremely useful in constructive modeling of heterogeneous objects, where both the geometry of the object and the material features are defined constructively using normal primitives and SARDF operations.

Our approach is a possible solution to the open question from [7] on computing the distance fields used to parameterize the distance distributions. We extend the constructive hypervolumes framework [10] to constructive design of shapes by signed distance functions, with direct applications in constructive heterogeneous modeling using the complete description of material design from [7].

Through the case studies, the viability of the proposed functions for constructive heterogeneous modeling is confirmed. The provided examples can be easily extended with more complex material distribution for the material features and more complex scheme for combination of the different attributes, since no restrictions prevent it in our model.

5.2 Extensions and research directions

In retrospect, one of the reasons for not using *min* / *max* in constructive modeling is the introduction of points of C^1 discontinuity. But as noticed earlier, such points may appear in the normal primitives due to the definition of the distance function. Some special treatment of the normal primitives is a source for future work.

We stated in the introduction that using *min* / *max* is keeping only an approximation of the exact distance field for the resulting function. Whether *min* or *max* is used, an approximation for the resulting distance field occurs in only one of the four quadrants, where the sharp edge of every contour line should be replaced by a circular arc with the opening of the arc given by the angle between the normals of both parameter shapes. The error between *min* / *max* or between the SARDF function and the exact distance field should be more carefully studied from the mathematical point of view.

The distance property of shapes and material features obtained by the proposed framework could be used to generate better quality mesh for finite element analysis using the algorithms for surface and volume discretization of FRep heterogeneous objects described in [39].

6 ACKNOWLEDGMENTS

The authors acknowledge the reviewers for fruitful comments. P.-A. Fayolle acknowledges support by Monbukagakusho, the Japanese ministry of Education, Culture, Sports, Science and Technology.

7 REFERENCES

[1] Ricci, A., 1973, "A Constructive Geometry for Computer Graphics", The Computer Journal, 16(2), pp.157–160.

[2] Sabin, M., 1968, "The Use of Potential Surfaces for Numerical Geometry", Technical Report VTO/MS/153.

[3] Kumar, V., and Dutta, D., 1997, "An Approach to Modeling Multi-Material Objects", Fourth Symposium on Solid Modeling and Applications, pp. 336-345.

[4] Kumar, V., Burns D., Dutta D., and Hoffman C., 1999, "A Framework for Object Modeling", Computer-Aided Design, 31(9), pp. 541-546.

[5] Bhashyam, S., Shin, K. H., and Dutta, D., 2000, "An Integrated Cad System for Design of Heterogeneous Objects", Rapid Prototyping Journal, 6(2), pp. 119-135.

[6] Chen, K., and Feng, X., 2003, "Computer-Aided Design Method for the Components Made of Heterogeneous Materials", Computer-Aided Design, 35(5), pp. 453-466.

[7] Biswas, A., Shapiro, V., and Tsukanov, I., 2004, "Heterogeneous Material Modeling with Distance

Fields", Computer Aided Geometric Design, 21(3), pp. 215-242.
[8] Requicha, A., 1980, "Representations for Rigid Solids: Theory, Methods, and Systems", ACM Computing Surveys, 12(4), pp.437-464.
[9] Nielson, G., 2000, "Volume Modelling", Volume Graphics, M. Chen, A. Kaufman, R. Yagel, Eds., Springer-Verlag, pp. 29-48.
[10] Pasko, A., Adzhiev, V., Schmitt, B., and Schlick, C., 2001, "Constructive Hypervolume Modeling", Graphical Models, 63(6), pp. 413-442.
[11] Pasko, A., Adzhiev, V., Sourin, A., and Savchenko, V., 1995, "Function Representation in Geometric Modeling: Concept, Implementation and Applications", The Visual Computer, 11(8), pp. 429-446.
[12] Rvachev, V., 1963, "On the Analytical Description of some Geometric Objects", Ukranian Academy of Sciences, 153(4), pp. 765-767.
[13] Qian, X., and Dutta, D., 2001, "Physics Based B-Spline Heterogeneous Object Modeling", In *ASME Design Engineering Technical Conference,* Pittsburgh.
[14] Jackson, T. R., 2000, "Analysis of Functionally Graded Material Object Representation Methods", PhD thesis, MIT, Ocean Engineering Department.
[15] Siu, Y. K., and Tan, S. T., 2002, "Modeling the Material Grading and Structures of Heterogeneous Objects for Layered Manufacturing", Computer-Aided Design, 34, pp. 705-716.
[16] Liu, H., Cho, W., Jackson, T. R., Patrikalakis, N. M., and Sachs, E. M., 2000, "Algorithms for Design and Interrogation of Functionally Gradient Material Objects", In *Proceedings of ASME 2000 IDETC/CIE 2000 ASME Design Automation Conference,* Baltimore, MD.
[17] Frisken, S., Perry, R., Rockwood, A., and Jones, T., 2000, "Adaptively Sampled Distance Fields: a General Representation of Shape for Computer Graphics", In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., pp. 249-254.
[18] Cohen-Or, D., Levin, D., and Solomovic, A., 1998, "Three-Dimensional Distance Field Metamorphosis", ACM Transaction on Graphics, 17(2), pp. 116-141.
[19] Jones, M. and Chen, M., 1994, "A New Approach to the Construction of Surfaces from Contour Data", Computer Graphics Forum, 13(3), pp. 75-84.
[20] Hart, J., 1996, "Sphere Tracing: A Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces", The Visual Computer, 12(10), pp. 527-545.
[21] Zhou, Y., Kaufman, A., and Toga, A., 1998, "3d Skeleton and Centerline Generation Based on an Approximate Minimum Distance Field", The Visual Computer, 14(7), pp. 303-314.
[22] Goldman, R. N., 1983, "Two Approaches to a Computer Model for Quadric Surfaces", IEEE Computer Graphics and Applications, 3(6), pp. 21-24.
[23] Satherley, R., and Jones, M., 2001, "Hybrid Distance Field Computation", Volume Graphics, pp. 195-209.
[24] Zhao, H., 2005, "A Fast Sweeping Method for Eikonal Equations", Mathematics of Computation, 74, pp. 603-627.
[25] Tsai, Y. R., 2002, "Rapid and Accurate Computation of the Distance Function Using Grids", Journal of Computational Physics, 178(1), pp. 175-195.
[26] Roessl, C., Zeilfelder, F., Nurnberger, G., and Seidel, H.-P., 2004, "Spline Approximation of General Volumetric Data", ACM Solid Modeling 2004.
[27] Sethian, J., 1999, *Level-Set Methods and Fast Marching Methods*. Cambridge University Press.
[28] Zhao, H., Osher, S., and Fedkiw, R., 2001, "Implicit Surface Reconstruction and Deformation Using the Level-Set Method", In *the 1st IEEE workshop on variational and level-set methods in computer vision, Canada.*
[29] Ohtake, Y., Belyaev, A., Alexa, M., Turk, G., and Seidel, H.-P., 2003, "Multi-Level Partition of Unity Implicits", ACM Transaction on Graphics, 22(3), pp. 463-470.
[30] Wang, M. Y., Wang, X., 2005, "A Level-Set Based Variational Method for Design and Optimization of Heterogeneous Objects", Computer Aided Design, special issue on "Heterogeneous Object Models and their Applications", 37(3), pp. 321-337.
[31] Biswas, A., and Shapiro, V., 2004, "Approximate Distance Fields with Non-Vanishing Gradients", Graph. Models, 66(3), pp. 133-159.
[32] Buchele, S. and Crawford, R., 2003, "Three-Dimensional Halfspace Constructive Solid Geometry Tree Construction from Implicit Boundary Representations", In *Proceedings of the eighth ACM symposium on Solid modeling and applications*, ACM Press, pp. 135-144.
[33] Shapiro, V. and Vossler, D., 1993, "Separation for Boundary to Csg Conversion", ACM Trans. Graph., 12(1), pp. 35-55.
[34] Barthe, L., Dodgson, N., Sabin, M., Wyvill, B., and Gaildrat, V., 2003, "Two-Dimensional Potential Fields for Advanced Implicit Modeling Operators", Computer Graphics Forum, 22(1), pp. 23-33.
[35] Hart, J., 1994, "Distance to an Ellipsoid", Graphics Gems IV, P. Heckbert, eds., Academic Press, Boston, pp. 113-119.
[36] Shin, K.-H., and Dutta, D., 2001, "Constructive Representation of Heterogeneous Objects", Journal of Computing and Information Science in Engineering, 1, pp. 205-217.
[37] Shepard, D., 1968, "A Two-Dimensional Interpolation Function for Irregularly Spaced Data", proceeding of the 23 National Conference, pp. 517-524.
[38] Rvachev, V. L., Sheiko, T. I., Shapiro, V., and Tsukanov, I., 2001, "Transfinite Interpolation Over Implicitly Defined Sets", Computer Aided Geometric Design, 18, pp. 195-220.
[39] Kartasheva, E., Adzhiev, V., Pasko, A., Fryazinov, O., and Gasilov, V., 2003, "Surface and Volume Discretization of Functionally Based Heterogeneous Objects", Journal of Computing and Information Science in Engineering, 3(4), pp. 285-294.

Fitting of Parameterized FRep Shape Models

Pierre-Alain Fayolle
Distributed and Parallel Processing Lab
University of Aizu
Fukushima-ken 965-8580
Japan
r5076501@u-aizu.ac.jp

Alexander Pasko
Faculty of Computer and Information Systems
University of Hosei
Tokyo 184-8584
Japan
pasko@k.hosei.ac.jp

Nikolay Mirenkov
Distributed and Parallel Processing Lab
University of Aizu
Fukushima-ken 965-8580
Japan
nikmir@u-aizu.ac.jp

Abstract

We introduce the use of parameterized Function Representation (FRep) for reverse engineering of constructive solids in CAD applications. Recovering the best parameters of a FRep model from the given scanned surface points turns out to be a difficult problem of nonlinear optimization. We recall the traditional methods for solving such problems: direct methods like Levenberg Marquardt or Newton and sampling methods like simulated annealing. In order to overcome problems of each method, we combine them in a two-step process. We apply and compare all these different methods to the recovery of some mechanical parts.

Keywords: function representation, reverse engineering, CAD, nonlinear optimization

1 Introduction

Solid modeling is extensively used in industry in various areas such as design, architecture, manufacturing. Nevertheless, there still remain objects which are not yet available as solid models. It is necessary to handle these objects in the same terms as other solid models. Therefore, special methods are needed for helping in the process of reverse engineering of such solids. In industry, reverse engineering can be very helpful in various common tasks like: maintenance, structure modification, revamping, and robot intervention.

In this paper, we restrict our work to the reverse engineering of constructive solids for CAD applications. We will not consider the problems arising in reverse construction of complex free-form objects, which are rather different. For recent progress in that domain, the reader can refer to the works by Ohtake et al [OBS03] and [OBA+03].

Traditional methods for reverse engineering of solid models rely on fitting some parametric [Fis02] or algebraic surfaces [BKV+02] to scan data, which is usually a cloud of points on or near the surface of the object. Fitting a parameterized model means finding the best set of parameters such that the model surface becomes as close as possible to the data points. We propose to use here parameterized Function Representation (FRep), introduced in [PASS95], as a model to be fitted. FRep models can represent various shapes and possess some good mathematical properties, making them suitable for recovery process.

In general, FRep models are non-linearly parameterized; fitting them to 3D data (3D points on or near the surface) should be done by nonlinear least square minimization of an error of a fit function. This error of fit gives a measure on how close the model is to the 3D points. The works [SP99] and [SS01] also deal with non-linear optimization of FRep models, but the optimization problems were not related to point cloud processing and the optimization criteria were formulated in different ways.

Direct methods for nonlinear fitting are available, but they may perform poorly in complex parameter spaces, and may be trapped into some local optima. We propose in this paper to solve this issue by combining direct methods with slower global sampling methods, that are less sensitive to the initial conditions and can escape local optima.

2 Parameterized FRep models

The function representation (FRep) was introduced in [PASS95] as a uniform representation of multidimensional geometric objects. An object (point set) in multidimensional space is defined by a single continuous real-valued function of point coordinates $F(\mathbf{x})$ which is evaluated at the given point by a procedure traversing a tree structure with primitives in the leaves and operations in the nodes of the tree. The points with $F(\mathbf{x}) \geq 0$ belong to the object, and the points with $F(\mathbf{x}) < 0$ are outside of the object. The geometric domain of FRep in 3D space includes solids with non-manifold boundaries and lower dimensional entities (surfaces, curves, points) defined by zero value of the function.

A primitive can be defined by an equation or by a "black box" procedure converting point coordinates into the function value. Solids bounded by algebraic surfaces, skeleton-based implicit surfaces, and convolution surfaces, as well as procedural objects (such as solid noise), and voxel objects can be used as primitives (leaves of the construction tree). Many operations such as set-theoretic, blending, offsetting, projection, non-linear deformations, metamorphosis, sweeping, hypertexturing, and others, have been formulated for this representation in such a manner that they yield continuous real-valued functions as output [PASS95], thus guaranteeing the closure property of the representation. R-functions originally introduced in [Rva63] provide C^k continuity for the functions exactly defining the set-theoretic operations. Because of this property, the result of any supported operation can be treated as the input for a subsequent operation, thus very complex models can be created in this way using a single functional expression.

An FRep model can be built in a constructive way with abstract parameters. The construction tree includes only specified operations and types of primitives, while the parameters of primitives and operations are not required and can be recovered using fitting. Generic models can exist for each particular application domain (application library of shapes) or should be created by the user.

In the rest of the paper, the notation $F(\mathbf{x}, \mathbf{a})$ is used for a parameterized FRep, where $\mathbf{x} = (x_1, x_2, x_3)$ is a point in the 3D space and $\mathbf{a} = (a_1, \ldots, a_m)$ is a vector of m parameters.

3 Fitting parameterized FRep: nonlinear least squares approach

3.1 Fitting problem formulation

The problem is to recover a solid from a set of 3D points, $S = \{\mathbf{x_1}, \ldots, \mathbf{x_N}\}$, scattered on the surface of the object. Given S, the task is to find the best configuration for the set of parameters $\mathbf{a}^* = (a_1, \ldots, a_m)$ so that the parameterized FRep model $F(\mathbf{x}, \mathbf{a}^*)$ closely fits the data points. $F(\mathbf{x}, \mathbf{a})$ is an FRep model, made in a constructive way, which approximates the shape of the solid being reverse engineered. The vector of parameters $\mathbf{a}$ controls the final shape of the solid and the best fitted parameters should give the closest possible model according to the information provided by S.

For computing how close a given point is to the surface of the solid with the current set of parameters, a fitness function is needed. The FRep model $F(\mathbf{x}, \mathbf{a})$ itself can serve for defining such an algebraic distance, because of the natural distance property of FRep models. The error of fit thus becomes simply the square of the defining function values at all points (the surface of the solid being the set of points with zero function value):

$$error(\mathbf{a}) = \frac{1}{2} \sum_{i=1}^{N} F^2(\mathbf{x_i}, \mathbf{a}) \qquad (1)$$

Now, we are searching for the vector of parameters $\mathbf{a}^*$ minimizing the error of fit from equation 1. We consider firstly direct methods for minimizing function with nonlinear parameters.

3.2 Nonlinear minimization of least squares

The best set of parameters $\mathbf{a}^*$ is found by minimization of the least square error (equation 1). This least square error is a nonlinear function of the parameters $\mathbf{a}$. Traditional methods for solving such problems are Levenberg-Marquardt methods [PFTV92], [Mor78] or modified Newton type methods [DGW81]. Algorithms of both types

proceed iteratively from an initial set of parameters and try to converge to a minimum in the parameter space. They first search for a privileged direction to go in the parameter space and then for a step to move in that direction. Levenberg-Marquardt (LM) and Modified Newton (MN) algorithms differ in the selection of direction and in the ways of computing the step. In our experiments, we use a modified version of a Newton type algorithm discussed in [DGW81] and available from Netlib[1].

3.3 Discussion of direct methods

Unfortunately, classical methods such as LM and MN can in general guarantee only a convergence to a local minimum. For some parameter spaces with complex topology like, for example, where multiple local minima exist, such methods are likely to be trapped into a local optimum and to stop there. Good initial parameters are very important, because they will determine to which minimum the algorithm may converge. Usually, if the parameters are not in the neighborhood of the global minimum, it is unlikely to converge to it, and the method instead will reach a local minimum.

It is possible with some further analysis of the model to have some additional information for getting better estimation of the starting parameters. It may also be possible to restart the algorithm with a different starting point to see if it comes back to the same minimum.

4 Simulated annealing for fitting parameterized FRep models

We propose here to use a sampling algorithm (simulated annealing, SA) as an alternative method for fitting a parameterized FRep model. Techniques based on simulated annealing have been proven to be efficient for solving global optimization problems with complex parameter space topology [PFTV92], [MRR$^+$53], and [KGV83].

4.1 General idea of simulated annealing

When trying to minimize an objective function usually only downhill steps are accepted, but in simulated annealing some uphill steps may be accepted.

The method of simulated annealing was inspired by the behavior of some thermodynamical processes. In liquids with high temperature, the molecules move freely with respect to one another. When the liquid is cooled down, the mobility of the molecules decreases, and finally they stop. If the cooling process is not too fast, the system finishes in a state of minimum energy.

The probability of accepting an uphill step is made by analogy with the Boltzmann law, which states that a system in thermal equilibrum has its energy distributed probabilistically among all different energy states. Even for a low temperature there is a chance for the system to be in a high energy, so that it can escape the local minimum energy and find better one.

The above observations were applied by Metropolis [MRR$^+$53] to numerical computation. The choice of a good cooling schedule is the difficult part of the algorithm [Haj88], we use in our experiments an implementation based on the algorithm described in [GFR94].

4.2 Discussion of simulated annealing

Methods based on simulated annealing may be a good alternative to classical direct methods in global optimum search among many local optima, but they are facing some major problem of efficiency: the objective function needs to be sampled a huge number of times before reaching convergence.

Because of that, simulated annealing seems to be a less interesting option for nonlinear fitting of FRep. A suitable solution would be to combine it with a more direct method, and to switch between the methods once the neighborhood of the global minimum has been reached; we explore this potential solution in the next section.

5 Hybrid fitting method

In this section, we propose an approach combining direct methods with sampling methods based on simulated annealing. By doing so, we wish to have a method being able to avoid local minima, but still remaining fast enough. We propose for that

[1] www.netlib.org

to distinguish two phases in the fitting process as follows:

- At the beginning, a sampling method based on simulated annealing is used to get a first fit of the model,
- Then, it switches to a direct method to extract the parameters that fit closely the model to the 3D point set.

This system presents some interesting characteristics compared to the methods from section 3 and 4. The first step should give a configuration in the parameter space being in the vicinity of the global minimum and thus should help avoiding local minima. The second step should guarantee a faster convergence and should also converge to a better fitted model than using a sampling algorithm alone.

At the beginning of the fitting process, the input consists of a preliminary selected or specially built parameterized FRep model, a set of data 3D points, and an initial parameters estimation. The values of the initial parameters are not so crucial, because the sampling SA algorithm can escape local minima. When the parameters have geometric interpretation, it is also possible to guess initial values, even if not accurate at all. In the case when parameters have less obvious meaning, like coefficients in blending operations, it is more difficult to provide good initial estimation, therefore, some random initial values in the parameter space may be chosen.

The switch to a direct method is performed when there are some indications that an acceptable fit has been reached: it can be when the least square error is under some given threshold or when the fitted model looks close enough to the point cloud (in the case the visual feedback to the user is provided). Then, the system switches to the second stage, when a traditional direct method is used. Since the region of the global minimum has already been determined by the first phase of the system, convergence to the global minimum should be quickly obtained, and local minima avoided.

The goal of the two-phase method described above is to provide a fitting system for parameterized FRep models that does not require a good initial configuration, can escape local minima, and has an acceptable convergence rate. The next section presents the current implementation and some experiments with the developed software system.

6 Experiments

6.1 A first mechanical part

The direct method MN (section 3), the sampling simulated annealing method (section 4) and the two-phase method (section 5) have been tested for the reverse engineering of a simple object, modeled for testing purposes using HyperFun[2], a set of tools for FRep geometric modeling. The surface of the object has been sampled to create the data set of 10714 3D points (Figure 1).

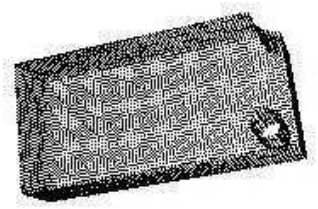


Figure 1: Data set used for the reverse engineering of the mechanical part. It contains 10714 3D points, scattered on the surface of the part.

The FRep defining function F shown below has been used as a parameterized model for the recovery process:

$$F(\mathbf{x},\mathbf{a}) := (box(\mathbf{x},\mathbf{a}) \backslash cylinderZ(\mathbf{x},\mathbf{a})) \backslash cylinderZ(\mathbf{x},\mathbf{a}); \qquad (2)$$

This FRep model consists of three simple primitives: one box and two infinite cylinders oriented along the Z axis, each primitive is defined by its parameterized model. For example, in the case of the cylinder, the defining function is: $cylinder(x,a) := a[1]^2 - (x[1]-a[2])^2 - (x[2]-a[3])^2$, where $a[1]$, $a[2]$, and $a[3]$ are parameters meaning the radius, and the center of the cylinder correspondingly. All the primitives are combined together using the R-subtraction operator ($\backslash$), which is itself defined analytically as discussed in [Rva63] and [PASS95].

Twelve parameters have been released in the model; they correspond to the lower-left corner of the box, three dimensions of the box, and the

[2] www.hyperfun.org

center and radius for each of the cylinders. The fitting algorithms need an initial estimation for the parameters, in the tests we used two sets of initial values configurations: one is close to the best fit ($set1 := \{-5, -4, -2, 10, 5, 4, 5, 3, 1, 3, -2, 1\}$), while the second one is a wrong set ($set2 := \{1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1\}$).

The results of the tests are given in terms of the following: least square error of the reconstructed model for the three methods MN, SA, and hybrid SAMN (Table 1), time given in seconds taken to converge to the best fit for each of these methods (Table 1), and the visual shape of the best fit (Figure 3).

	Time in sec		Least square error	
	set1	set2	set1	set2
MN	1.852	9.643	5.47	595.04
SA	1635.09	1773.109	5.49	5.49
SAMN	72.042	144.177	5.47	5.47

Table 1: Time (in seconds) taken by each method to converge to the best fit and Least square error (sum of the deviations squared) of the best fit for each set of initial values.

From Table 1, it appears that the direct method stops in a local minimum for the set2 of initial parameters, resulting in a wrong shape (Figure 3, right shape), whereas the simulated annealing always converges to the global minimum. Unfortunately, the counterpart is the slow rate of convergence for the sampling method (Table 1). SA turns out to be 200 times slower.

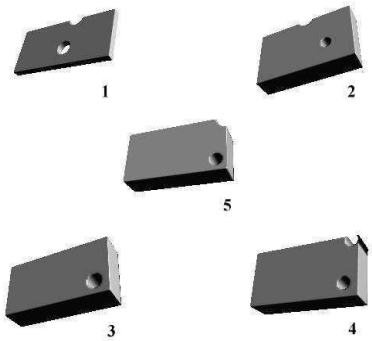


Figure 2: Evolution of the shape during the fitting process using hybrid method.

When using a combination of SA and MN, we can always recover the good parameters and the good shape (Figure 3, left shape, and Table 1, last line). See the steps of the shape evolution during the hybrid method search in Figure 2. The overhead in speed is in magnitude of 10 compared to the direct method (MN), but local optimum and bad shape recovery is avoided. Compared to SA it is between 10 and 20 times faster.

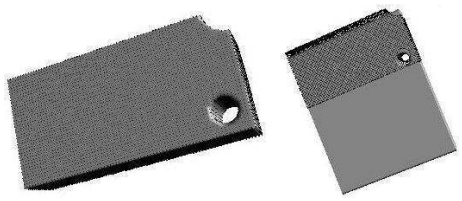


Figure 3: Shapes for the best fitted FRep in two cases: (right) best fitted, but wrong, model when starting with set2 and using the MN method; (left) best fitted model when starting with the set2 and using the hybrid method.

6.2 A second mechanical part

A second set of tests has been performed on a more complex mechanical part. The point-set for the corresponding surface is shown Figure 4.

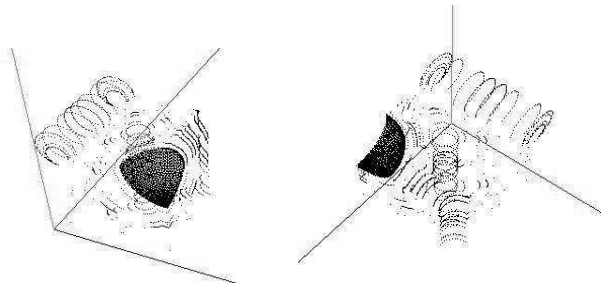


Figure 4: Two views of the point set consisting of points on the surface of the mechanical part and used for fitting of the FRep model parameters.

The FRep model including 12 parameters was sketched corresponding to the visual shape of the model. The initial values for the parameters were chosen randomly, although some of them could have been easily guessed. The convergence was obtained by the hybrid approach described in Section 5. The result in terms of the fitting function value and the mean error is presented in Table 2.

The FRep shape corresponding to the best set of parameters is shown in Fig. 5.

fit function	0.667
mean error	0.011622

Table 2: Fitting function value for the best fit set of parameters and the corresponding mean error.

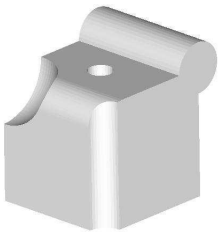


Figure 5: The shape defined by the FRep with the best fitted set of parameters.

7 Conclusion

We have introduced in this paper the use of parameterized FRep for reverse engineering of constructive solid models from 3D data sets. Parameterized FRep presents several advantages over the traditional parametric models for reverse engineering: uniform representation of shapes of different topology and dimensions, rich system of primitives and operations, support of models with heterogeneous internal structure, the error of fit function is naturally defined by the FRep function itself, because of its distance property.

For fitting an FRep model to 3D data set, numerical methods for solving nonlinear optimization problem are required. We have discussed and tested first the classical direct methods relying on the computation of a privileged direction in the parameter space. It was found that these methods require a good initial estimation in order to converge to the best fit. Otherwise, only a convergence to a local optimum could be guaranteed.

Then, we have discussed and tested simulated annealing methods as a potential alternative to direct methods. Such sampling methods have been proven to converge to the global minimum and escape local minima, even in parameter space with complex topology. Their convergence seemed also to be less dependent of the choice of a good initial configuration of parameters. However, SA methods suffer from a low convergence rate, which makes them a less attractive solution than direct methods.

Finally, we have combined both methods, in order to overcome their weaknesses. The two-phase process approach has been applied to a simple mechanical part, available under the form of a 3D point set, and compared with the methods described above. The result is that the hybrid method is faster than simulated annealing and converging to the best fit, even with poor initial parameters. A more complex mechanical part has been also considered and recovered using the hybrid method.

Future works include trying genetic algorithms instead of simulated annealing to see if the overhead in the sampling method can be decreased, recovering more complex constructive objects and heterogeneous objects.

References

[BKV+02] P. Benko, G. Kos, T. Varady, L. Andor, and R. Martin, *Constrained fitting in reverse engineering*, Computer Aided Geometric Design **19** (2002), 173–205.

[DGW81] J.E. Dennis, D.M. Gay, and R.E. Welsch, *An adaptative nonlinear least-squares algorithm*, ACM Transaction on mathematical software **7** (1981), 348–368.

[Fis02] R. Fisher, *Applying knowledge to reverse engineering problems*, Proceedings of Geometric Modeling and Processing, 2002, pp. 149–155.

[GFR94] W. Goffe, G. Ferrier, and J. Rogers, *Global optimization of statistical functions with simulated annealing*, Journal of econometric **60** (1994), 65–99.

[Haj88] B. Hajek, *Cooling schedules for optimal annealing*, Mathematics of Operations Research **13** (1988), 311–329.

[KGV83] S. Kirkpatrick, C. Gelatt, and M. Vecchi, *Optimization by simulated anneal-*

ing, Science **220, 4598** (1983), 671–680.

[Mor78] J. Moré, *The levenberg-marquardt algorithm implementation and theory*, Lecture notes in mathematics No630 Numerical analysis, vol. 630, New-York, 1978, pp. 105–116.

[MRR+53] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, *Equations of state calculations by fast computing machine*, J. Chem. Phys. **21** (1953), 1087–1092.

[OBA+03] Yutaka Ohtake, Alexander Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel, *Multi-level partition of unity implicits*, Proceedings of SIGGRAPH 2003, ACM, 2003.

[OBS03] Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel, *A multi-scale approach to 3d scattered data interpolation with compacly supported radial basis functions*, Proceedings of Shape Modelling 2003, 2003.

[PASS95] Alexander Pasko, Valery Adzhiev, Alexei Sourin, and Vladimir Savchenko, *Function representation in geometric modeling: concepts, implementation and applications*, The visual Computer **11** (1995), 429–446.

[PFTV92] W. Press, B. Flannery, S. Teukolsky, and W. Vatterling, *Numerical recipes in c - the art of scientific computing*, Cambridge University Press, Cambridge, 1992.

[Rva63] V. Rvachev, *On the analytical description of some geometric objects*, Reports of Ukranian Academy of Sciences **153** (1963), 765–767.

[SP99] V. Savchenko and A. Pasko, *Evolutionary optimization of functionally defined shapes: case study of natural optical objects*, Proceedings of Computer Graphics International, 1999, pp. 20–24.

[SS01] V. Savchenko and L. Schmitt, *Reconstructing occlusal surfaces of teeth using a genetic algorithm with simulated annealing type selection*, Proceedings of ACM Symposium on Solid Modeling and Application, 2001, pp. 39–46.

Shape recovery using functionally represented constructive models

Pierre-Alain Fayolle*, Alexander Pasko**, Elena Kartasheva***, and Nikolay Mirenkov*

*University of Aizu, Fukushima-ken 965-8580, Japan

**Hosei University, Tokyo 184-8584, Japan

***Institute for Mathematical Modeling, Russian Academy of Science, Moscow, Russia

d8052103@u-aizu.ac.jp, pasko@k.hosei.ac.jp, ekart@imamod.ru, nikmir@u-aizu.ac.jp

Abstract

We propose a method which helps to fit existing parameterized function representation (FRep) models to a given dataset of 3D surface points. Best fitted parameters of the model are obtained by using a hybrid algorithm combining simulated annealing and Levenberg-Marquardt methods. The efficiency of the approach is shown for recovery of two test items. We show through the CAD model processing an application of the proposed approach to the shape recovery followed by finite element mesh generation and adaptation.

Keywords: *shape recovery, function representation, fitting, non-linear optimization, finite element meshes*

1. Introduction

One of the actual problems in solid modeling is dealing with objects which are not yet (or not anymore) available as solid models. It is necessary to handle them in the same terms as other solid models. Therefore, special methods are needed for reverse engineering such solids. In this paper, we restrict our work to the shape recovery of constructive solids with smooth surfaces for cultural heritage and finite element meshes (FEM) applications.

One of the goals is to have models that can be later reused for modeling, modifications, or analysis. For instance, example-based modeling techniques are discussed for the case of human body in [14].

Previous work Reverse engineering of solid models relies on fitting some mathematical models, traditionally parametric or algebraic surfaces [2], [4], to scan data. Fitting a model consists in finding the best set of parameters such that the model becomes as close as possible to the data points. The idea of knowledge based reverse engineering was introduced in [4] and [2]. Relations between parameters or objects are used to guarantee the production of models with sufficient accuracy reproducing symmetry or alignment. This interpretation of shape recovery well suits boundary representation with segmented point clouds. The main problem with this approach comes from the difficulty to extend the set of allowable shapes, because a corresponding segmentation would be required, which can be difficult or even impossible in the case of complex blends or sweeps. Furthermore, it may be difficult for the resulting model, generally available as a Brep, to be reused in extended modeling or analysis.

We use a different interpretation of shape recovery and a different model, the function representation of objects [12]. In our approach, standard shapes and relations are interpreted as primitives and operations of a constructive model. The input information provided by the user is a template (sketch) model, where the construction tree contains only specified operations and types of primitives while the parameter values of operations and primitives are not defined and are recovered by fitting. In general, FRep models are non-linearly parameterized and fitting them to 3D data should be done by non-linear least square minimization of a fit function, which indicates how close the model is to the 3D points.

2. The developed method

2.1 Parameterized function representation

The function representation (FRep) was introduced in [12] as a uniform representation of multidimensional geometric objects. An object (point set) is defined by a single continuous real-valued function of point coordinates $f(\mathbf{x})$ which is evaluated at the given point by a procedure traversing a tree structure with primitives in the leaves and operations in the nodes of the tree. The points with $f(\mathbf{x}) \geq 0$ belong to the object, and the points with $f(\mathbf{x}) < 0$ are outside of the object.

An FRep model can be built in a constructive way using primitives and operations with abstract parameters.

The modification of these parameters can result in various shapes, which can also be tuned to fit some modeling criteria. In the rest of the paper, the notation $f(\mathbf{x}, \mathbf{a})$ is used for a parameterized FRep model, where $\mathbf{x} = (x_1, x_2, x_3) \in \mathbb{R}^3$ is a point in the 3D space and $\mathbf{a} = (a_1, \ldots, a_m) \in \mathbb{R}^m$ is a vector of m parameters.

2.2 Origin of the template model

Template model can exist in specialized libraries for each application domain (mechanical design, human prosthesis design, and others) and may be reused, or need to be created by the user. In the latter case, a modeling work needs to be done by a designer. A parameterized model can be created using measurements or scans of a typical object. The model is required to keep basic ratios of the measured sample object and to proportionally change the dependent parameters according to introduced constraints. In case of scanned data available for a typical object, fitting of the template parameters can be also employed to establish basic ratios and constraints.

2.3 Parameters estimation

Given $S = \{\mathbf{x_1}, \ldots, \mathbf{x_N}\}$, a set of 3D points scattered on the surface of the object, the task is to find the best configuration for the set of parameters $\mathbf{a}^* = (a_1, \ldots, a_m)$ so that the parameterized FRep model $f(\mathbf{x}, \mathbf{a}^*)$ closely fits the data points. The vector of parameters $\mathbf{a}$ controls the final shape of the solid and thus the best fitted parameters should give the closest possible model according to the information provided by S.

In order to evaluate the differences between the point set, and the surface of the solid for the current vector of parameters, a fitness function is needed. The function f itself defines an algebraic distance between the current point of evaluation $\mathbf{x}$ and the surface $f(\mathbf{x}) = 0$ [12]. Therefore, the error of fit can be formulated as follows:

$$error(\mathbf{a}) = \frac{1}{2} \sum_{i=1}^{N} f^2(\mathbf{x_i}, \mathbf{a}) \qquad (1)$$

When existing, primitives defined by Euclidean distance functions should be preferred, as it is noticed in [4]. A list of existing primitives with known Euclidean distance functions is given in [6].

The estimation of the parameters minimizing the objective function Eq. 1 is done by non-linear optimization and described in the following subsection.

2.4 Numerical optimization

The most common local methods for solving problems of non-linear optimization are Levenberg-Marquardt [11] or Newton type [3] methods. Unfortunately, such methods need to be started in the neighborhood of the solution to avoid local optima. A local optimum results in a wrong reconstructed shape, as seen in Fig. 1. It is usually difficult

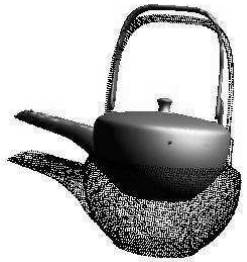


Figure 1. Local minimum effect: result of the fitting with a local method stopping in a local optimum. The discrete data points of the sake pot are also displayed for comparison.

to predict local optima, and even a good looking shape, like for instance in Fig. 1, may be incorrectly fitted.

To escape local minima and to avoid the problem of the initial estimation of parameters, a stochastic algorithm may be considered: we use simulated annealing [8, 10] in our experiments. It should be noticed that in some cases the initial estimation of the parameters may be difficult or even impossible to get, for example, with parameters of blending operations. In our experiments, the initial parameters are chosen randomly.

However, a stochastic method is a time consuming process, because it requires a huge number of evaluation of the objective function. Even once in the neighborhood of the solution, the process remains slow, sampling the objective function in the parameter space. Therefore, it seems reasonable to start with a stochastic method and combine it with a gradient method [1]. It accelerates the convergence and enhances the final accuracy. In our experiment, the switch between both methods is done according to a combination of a threshold value for the fitness function (Eq. 1) specified by the user and a visual feedback of the current shape and the set of points.

3 Experiments

3.1 Lacquer ware sake pot

The first example is the fitting of a model of a handcrafted lacquer ware pot, which is used for pouring sake (Japanese rice wine). The discrete data set of the sake pot includes 27048 3D points, scattered on the surface of the object. The parameterized model of the sake pot sketched

and discussed in the work on cultural heritage [13] is reused in our experiment. The parameterized model was created using hand measurements of a typical sake pot. The major parameters are the coordinates of the origin (position), the basic radius of the pot body, and the height of the pot handle. The model is required to keep basic ratios of the measured sample object and to proportionally change the dependent parameters like those of the blend area between the spout and the body, and the shape of the lid holder (note non-linear chages of these shapes in Fig. 2).

In the test, a value of 1000 for the fit function is used as a threshold value to determine the switch to a local method. This value corresponds to an average error of 0.04 of the fit function (Eq. 1), which we consider small enough in order to escape local minima and confirmed by visual feedback. This threshold is reached after 344 seconds on a Pentium IV PC. Then, the obtained parameters are reused as initial values for the local Quasi-Newton method. The steps of the evolution of the shape during the hybrid fitting of the FRep model can be seen at Fig. 2.

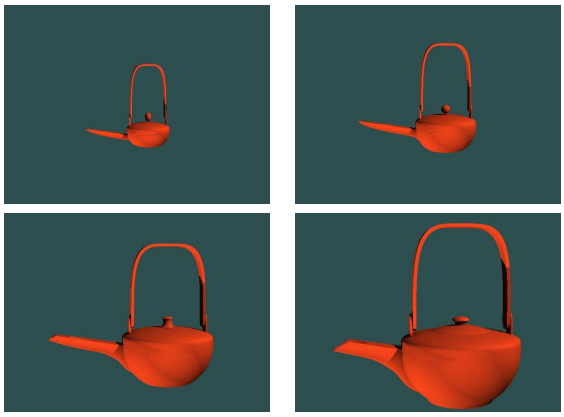


Figure 2. Evolution of the sake pot shape during the fitting process

3.2 Application in Finite Element Meshes (FEM)

Approach to FEM generation Surface remeshing is very important for applications associated with numerical simulation procedures, in particular with finite element analysis (FEA). These applications impose strict constraints on the quality of the surface approximation and the shapes and sizes of mesh elements. Moreover finite element meshes have to be adapted both to physical and geometric features of computational tasks. Changes in the boundary or initial conditions of the simulated process may cause remeshing even if the computational domain remains the same. In many cases the initial description of computational domains in FEA is represented by their boundary surface triangulations. These triangulations can be exported from various modeling systems, produced by 3D scanning, or be a result of previous FE computations. Usually these initial triangulations consist of badly shaped triangles and are not adapted to physical conditions and an appropriate remeshing is required. Mesh refinement and optimization procedures need accurate information about the geometry of the computational domain. Therefore, the creation of an adequate description of a solid based on the initial triangulation of its boundary surface is an important problem for the FE mesh generation and optimization. Different approaches were considered to solve this problem. In [5], finite element adaptation is based on the local approximation of the underlying surface geometry by a quadric surface. The authors of [9] convert a CAD model into a volume representation by sampling its distance field on a uniform grid and then applying the extended marching cubes algorithm to this volume.

Taking into account that many mechanical parts can be represented as constructive solids we propose to apply FRep recovery to support FE mesh generation for objects whose initial geometry is represented by boundary surface triangulations. The initial mesh is used for the selection or creation of a parameterized FRep model. Then, the parameters of the FRep model are fitted to the vertices of the mesh. The final model can be used for the surface and volume finite element adaptation by the methods described in [7].

Fitting to a CAD mesh As an example of application of the FRep shape recovery for the FEM generation, a parameterized FRep model corresponding to the CAD mesh Fig. 3 (top, left) is created and fitted using the previously proposed techniques.

fit function	0.667
mean error	0.011622

Table 1. Fitting function value for the best fit set of parameters and the corresponding mean error.

The FRep model including 14 parameters is sketched corresponding to the shape shown in Fig. 3, top, left; the initial values for the parameters are chosen randomly.

The convergence is obtained using the method proposed in the paper, and the results in terms of the fitting function value and the mean error are presented in Table 1. The FRep shape corresponding to the best set of parameters is shown in Fig. 3, top, right.

Starting with the acquired FRep model, it is then possible to apply the mesh adaptation methods from [7]. The results of such methods are shown in Fig. 3, bottom. The

left picture shows an optimized surface mesh, which was then used for the 3D tetrahedral mesh generation (right) using the extended advancing front method [7].

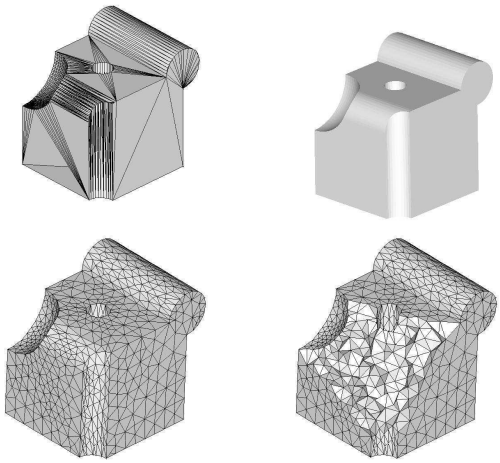


Figure 3. A surface mesh, generated by a CAD system (top, left), the recovered shape (top, right), the associated optimized mesh (bottom, left), and the 3D tetrahedral mesh generated from it (bottom, right).

4 Conclusion

We introduced an application of parameterized FRep models for shape recovery of constructive solid models from 3D point clouds. The use of parameterized FRep models presents some advantages over the traditional reverse engineering approach: no segmentation of the data set is required, a quite extensive and extensible set of FRep primitives and operations exist and can be utilized, all the primitives are defined by ready to use algebraic functions or procedures, and an existing special high-level language is available for the model representation.

The fitted template model can be analyzed, classified, or reused in other applications, for example, in further remodeling, animation, or rapid prototyping. Some of these applications may be more difficult or impossible to implement with Brep models, which are the results of traditional reverse engineering methods.

Nevertheless, the proposed approach revealed also some existing problems. A generic parameterized model used for the fitting has to be available or to be created by the user. The evaluation of the defining function for a complex shape can be also time-consuming, which results in a time consuming optimization process.

There are still ways to improve the presented approach. The semi-automatic creation of generic models can be envisaged with application of genetic algorithms and genetic programming. The problem of the parameterized model adequacy evaluation has to be seriously considered when using these methods. The global fitting method can also be enhanced by replacing simulated annealing with genetic algorithms.

Acknowledgements

P.-A. Fayolle acknowledges support by Monbusho, the Japanese Ministry of Education and Research.

The authors would like to thank the anonymous reviewers for their valuable comments.

References

[1] A. Alcolea, J. Carrera, and A. Medina. A hybrid marquardt simulated annealing method for solving the groundwater inverse problem. In *Calibration and reliability in groundwater modeling*, number 265, pages 157–163. IAHS, 2000.

[2] P. Benko, G. Kos, T. Varady, L. Andor, and R. Martin. Constrained fitting in reverse engineering. *Computer Aided Geometric Design*, 19:173–205, 2002.

[3] J. Dennis, D. Gay, and R. Welsch. An adaptative nonlinear least-squares algorithm. *ACM Transaction on mathematical software*, 7:348–368, 1981.

[4] R. Fisher. Applying knowledge to reverse engineering problems. In *Proceedings of Geometric Modeling and Processing*, pages 149–155, 2002.

[5] P. Frey and H. Borouchaki. Geometric surface mesh optimization. *Computing and visualization in science*, 1(3):113–121, 1998.

[6] J. Hart. Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer*, 12(10):527–545, 1996.

[7] E. Kartasheva, V. Adzhiev, A. Pasko, O. Fryazinov, and V. Gasilov. Discretization of functionally based heterogeneous objects. In G. Elber and V. Shapiro, editors, *ACM Symposium on solid modeling and applications*, pages 145–156. ACM, 2003.

[8] S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[9] L. Kobbelt, M. Botsch, U. Schwanecke, and H.-P. Seidel. Feature sensitive surface extraction from volume data. In *Procedings of SIGGRAPH 2001*, pages 57–66. ACM, 2001.

[10] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computing machine. *J. Chem. Phys.*, 21:1087–1092, 1953.

[11] J. Moré. The levenberg-marquardt algorithm implementation and theory. In *Lecture notes in mathematics No630 Numerical analysis*, volume 630, pages 105–116. New-York, 1978.

[12] A. Pasko, V. Adzhiev, A. Sourin, and V. Savchenko. Function representation in geometric modeling: concepts, implementation and applications. *The visual Computer*, 11:429–446, 1995.

[13] G. Pasko, A. Pasko, C. Vilbrandt, and T. Ikedo. Virtual shikki and sazaedo: shape modeling in digital preservation of japanese lacquer ware and temples. In R. Durikovic and S. Czanner, editors, *Spring Conference on Computer Graphics SCCG2001*, pages 147–154. IEEE, 2001.

[14] H. Seo and N. Magnenat-Thalmann. An example based approach to human body manipulation. *Graphical Models*, 66(1):1–23, 2004.

3D Shape Reconstruction of Template Models Using Genetic Algorithms

P.-A. Fayolle[1], C. Rosenberger[2], C. Toinard[3]

[1] University of Aizu, Software Department
AizuWakamatsu, Fukushima ken 965-8580, Japan
email: d8052103@u-aizu.ac.jp
[2] Laboratoire Vision et Robotique, UPRES EA 2078
ENSI de Bourges - Université d'Orléans
10 boulevard Lahitolle, 18020 Bourges, France
[3] Laboratoire d'informatique Fondamentale d'Orléans, CNRS FRE 2490
ENSI de Bourges - Université d'Orléans
10 boulevard Lahitolle, 18020 Bourges, France

Abstract

We present in this communication a method, which enables to fit a 3D object defined by a functional representation (FRep) to a dataset of 3D points on its surface. A parametric FRep model sketching the point-set is fitted to the point-set. The best fitted parameters of the model are obtained by using a genetic algorithm, well known for its interesting properties in non-linear optimization. The efficiency of the approach is illustrated for reverse engineering applications.

Keywords: 3D reconstruction, reverse engineering, genetic algorithm, CAD, 3D visualization.

1. Introduction

Solid modeling is extensively used in industry in various areas such as design, architecture and manufacturing. Nevertheless, there still remains objects which are not yet available as solid models. It is necessary to handle these objects in the same terms as other solid models. Therefore, special methods are needed for helping in the process of reverse engineering of such solids. In industry, reverse engineering can be very helpful in various common tasks like: maintenance, structure modification, revamping and robot intervention.

In this paper, we restrict our work to the reverse engineering of constructive solids for CAD applications. We will not consider the problems arising in reverse construction of complex free-form objects, which are rather different. For recent progress in that domain, the reader can refer to the work by Ohtake et al [10].

Traditional methods for reverse engineering of solid models rely on fitting some parametric [4] or algebraic surfaces [1] to scan data (usually a cloud of points on or near the surface of the object). Fitting a parametrized model means finding the best set of parameters such that the model becomes as close as possible to the data points. We propose to use here parameterized function representation of objects, introduced in [11], as a model to be fitted. In our approach standard shapes and relations are interpreted as primitives and operations of a constructive model. The input information provided by the user is a template (sketch) model, where the construction tree contains only specified operations and types of primitives while the parameters of operations and primitives are not required and are recovered by fitting. Template models can exist for each domain of applications and may be reused, or needs to be created by the user.

In general, FRep models are non-linearly parameterized; fitting them to 3D data should be done by non-linear least square minimization of an error of a fit function, which gives a measure on how close the model is to the 3D points. To solve this problem, a stochastic search (such as Simulated Annealing) combined with a gradient method can be used (see [3]). We propose to explore here genetic algorithms as a global minimizer.

2. Developed Method

2.1. The parameterized function representation of objects

The function representation (FRep) [11] follows a constructive approach for modelling multidimensional objects. In the constructive modelling approach, a complex object is decomposed into a set of simple ones, called primitives, that are then combined by different operations. The associated data structure is called a construction tree. In the FRep model, the primitives are any continuous real valued functions, for example skeletal implicit surfaces, algebraic surfaces, ...The operations themselves are defined analytically in such a manner that they yield continuous real valued functions as output, ensuring the closure property of the representation.

In the FRep model, a multi-dimensional object is defined by a single continuous real valued function f. The points x for which $f(x) \geq 0$ belongs to the object, whereas the points for which $f(x) < 0$ are outside.

In the construction tree only specified operations and primitives are included, while the parameters of both parameters and primitives are tuned to some modelling criteria. In the rest of the paper, the notation $f(\mathbf{x}, \mathbf{a})$ is used for a parameterized FRep, where $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$ is a point in 3D space and $\mathbf{a} = (a_1, \ldots, a_m) \in \mathbb{R}^m$ is a vector of m parameters.

2.2. Model estimation

The problem is to recover a solid from a set of 3D points, $S = \{\mathbf{x_1}, \ldots, \mathbf{x_N}\}$, scattered on the surface of the object. Given S, the task is to find the best configuration for the set of parameters $\mathbf{a}^* = (a_1, \ldots, a_m)$ so that the parameterized FRep model $f(\mathbf{x}, \mathbf{a}^*)$ closely fits the data points. The FRep model $f(\mathbf{x}, \mathbf{a})$, built in a constructive way, approximates the shape of the solid being reverse engineered. The vector of parameters $\mathbf{a}$ controls the final shape of the solid and the best fitted parameters should give the closest possible model according to the information provided by S.

In order to evaluate the differences between the discrete representation of the solid, given by S, and the surface of the solid for the current vector of parameters, implicitly defined by f, a fitness function is needed. The function f defines an algebraic distance between the current point of evaluation $\mathbf{x}$ and the surface it implicitly defines. The error of fit thus becomes the sum of the square of the defining function values at all points of S (the surface of the solid being the set of points with zero function value):

$$error(\mathbf{a}) = \frac{1}{2} \sum_{i=1}^{N} f^2(\mathbf{x_i}, \mathbf{a}) \qquad (1)$$

Now, we are searching for the vector of parameters $\mathbf{a}^*$ minimizing the error of fit 1. This function is non-linear in $\mathbf{a}$.

2.3. Numerical optimization

Local methods for solving problems of non-linear optimization are Levenberg-Marquardt methods [9] or modified Newton type methods [2]. Both methods proceed iteratively from an initial vector of parameters and try to converge to a minimum in the parameter space. Unfortunately this initial vector needs to be in the neighborhood of the solution, otherwise the algorithm may stop in a local optimum, resulting in a wrong reconstructed shape.

To escape local minima, some sampling algorithms, like for example simulated annealing can be used. Techniques based on simulated annealing have been proven to be efficient for solving global optimization problems with complex parameter space topology [7]. Unfortunately such methods are computationally intensive, and very slow in the last steps of convergence, i.e. in the vicinity of the solution. Therefore, it seems reasonable to use simulated annealing as a starting process to go to the vicinity of the global minimum, and then switch to a local method. This approach has been considered in [3] for non-linear optimization of shape recovery. Nevertheless simulated annealing presents some limitations, [12] reports some problems with high dimensionality of the search space.

We explore in this communication the use of a genetic algorithm as an alternative.

2.4. Genetic algorithms

Genetic algorithms (GA) determine the optimal value of a criterion by simulating the evolution of a population until survival of best fitted individuals [5]. Survivors are individuals obtained by crossing-over, mutation and selection of individuals from the previous generation. A genetic algorithm is summarized below, and details on each steps are described afterward:

1. definition of the initial population and computation of the fitness function of each individual,
2. selection of individuals,
3. mutation and crossing-over of individuals,
4. evaluation of individuals in the population,
5. back to the step 2 if the stopping criterion is not satisfied.

The use of a genetic algorithm requires the determination of some fundamental issues: the genotype, the initial population, the fitness function, the operators on the genotypes, and the termination criterion. Each of these issues is described below:

1. *genotype* : parameters of the model to estimate,
2. *initial population* : a set of random individuals,
3. *fitness function* : this function quantifies the fitness of an individual to the environment by considering its genotype. In our case, it corresponds to the function defined by the equation (1),
4. *operators on genotypes* : they define alterations on genotypes in order to make the population evolve during generations. Three types of operators are used :
   - individual mutation : genes of an individual are modified in order to better adapt to the environment. We use the Non-Uniform mutation process which randomly selects one chromosome x_i, and sets it equal to a non-uniform random number :

$$x_i' = \begin{cases} x_i + (b_i - x_i)f(G) & if\ r_1 < 0.5 \\ x_i - (x_i + a_i)f(G) & if\ r_1 \geq 0.5 \end{cases} \quad (2)$$

     where
     $f(G) = (r_2(1 - \frac{G}{G_{max}}))^b$
     r_1, r_2 : uniform numbers in the interval [0,1]
     a_i, b_i : lower and upper bound of chromosome x_i
     G : the current generation
     G_{max} : the maximum number of generations
     b : a shape parameter
   - selection of an individual : individuals that are not adapted to the environment do not outlive to the next generation. We used the normalized geometric ranking selection method which define a probability P_i for each individual i to be selected:

$$P_i = \frac{q(1-q)^{r-1}}{1-(1-q)^n} \quad (3)$$

     where
     q : the probability of selecting the best individual
     r : the rank of individual, where 1 is the best
     n : the population size
   - crossing-over : two individuals can reproduce by combining their genes. We use the arithmetic crossover which produces two complementary linear combinations of the parents :

$$\begin{aligned} X' &= aX + (1-a)Y \\ Y' &= (1-a)X + aY \end{aligned} \quad (4)$$

     where
     X,Y : genotype of parents
     a : a uniform number in the interval [0,1]
     X',Y' : genotype of the linear combinations of the parents
5. *stopping criterion* : We choose to consider the stability of the standard deviation of the evaluation criterion of the population.

Following [8], we use a real-valued encoded genotype, with operators on genotype chosen in consequence to work on real-valued encoded genotypes. This study proved than real-valued GA is an order of magnitude more efficient than binary GA in term of CPU, and offers a higher precision.

Genetic algorithms present attractive properties for the estimation of the FRep parameters model. No initial solution is needed, in contrary to the local methods. They have the ability to proceed with a high dimensional and complex search space. Finally, a parallel implementation can be easily done, due to the nature of the genetic algorithm [8].

3. Experimental results

In all the tests, a ©Matlab implementation of a genetic algorithm is used [6].

3.1. First example: a first synthetic CAD part

We tested first the developed approach for the case of reverse engineering of a simple object, modelled for testing purposes using HyperFun [1], a set of tools for FRep geometric modelling. The surface of the object has been sampled to create the data set of 10714 3D points (Figure 1).

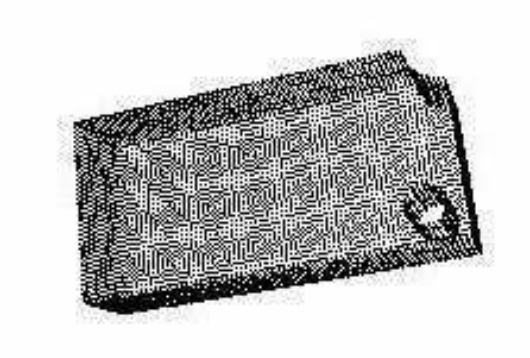


Figure 1. Data set used for the reverse engineering of the mechanical part.

The FRep F below was used as a parameterized model for the recovery process:

$$F(\mathbf{x}, \mathbf{a}) := (box(\mathbf{x}, \mathbf{a}) \backslash cylinderZ(\mathbf{x}, \mathbf{a})) \backslash cylinderZ(\mathbf{x}, \mathbf{a}); \quad (5)$$

This model consists of three simple primitives: one box and two infinite cylinders oriented along the Z axis, each

1 www.hyperfun.org

primitive is defined by its parameterized model. For example, in the case of the cylinder, the defining function is: $cylinder(x, a) := a[1]^2 - (x[1] - a[2])^2 - (x[2] - a[3])^2$, where $a[1]$, $a[2]$, and $a[3]$ are parameters meaning the radius, and the center of the cylinder correspondingly. All the primitives are combined together using the R-subtraction operator ($\setminus$), which is itself defined analytically as discussed in [11]. We let the algorithm run for 200 iterations, starting from a population of 2000 individuals. The best individual found has a least square error of 3.13.

3.2. Second example: a real-world mechanical part

The second test has been performed on a mechanical part (see the point-set for the corresponding surface in Fig. 2).

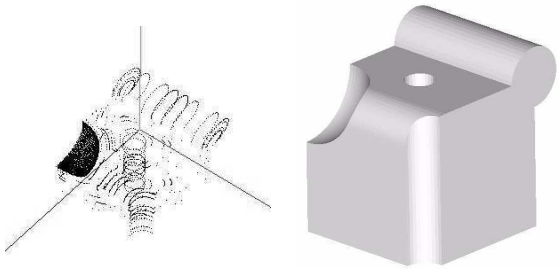


Figure 2. Points on the surface of the mechanical part and the resulting FRep model. This mechanical data set is courtesy of E. Kartasheva.

The FRep model including 12 parameters was sketched corresponding to the visual shape of the model. The result in terms of the fitting function value is equal 3.03. We used 20000 individuals in the initial population and 500 iterations to converge to the final set of parameters resulting in shape visualized in Figure 2.

4. Conclusion

We have introduced in this paper the use of parameterized FRep for reverse engineering of constructive solid models from 3D data sets. Parameterized FRep presents several advantages over the traditional parametric models for reverse engineering: uniform representation of shapes of different topology and dimensions, rich system of primitives and operations.

For fitting an FRep model to 3D data set, we proposed a new alternative by using GA for solving non-linear optimization problem. The advantages of GA are to be able to proceed with a very important search space and it does not require an initial solution.

Future works concern the use of GA to obtain automatically the FRep model and its parameters to recover more complex constructive objects and heterogeneous objects.

Acknowledgement

C. Toinard and C. Rosenberger would like to thank financial support provided by the Conseil Général du Cher.

P.A. Fayolle acknowledges support by Mombusho, the Japanese ministry of research and education.

The authors acknowledge the reviewers for their valuable comments. The second model is courtesy of Elena Kartasheva.

References

[1] P. Benko, G. Kos, T. Varady, L. Andor, and R. Martin. Constrained fitting in reverse engineering. *Computer Aided Geometric Design*, 19:173–205, 2002.

[2] J.E. Dennis, D.M. Gay, and R.E. Welsch. An adaptative nonlinear least-squares algorithm. *ACM Transaction on mathematical software*, 7:348–368, 1981.

[3] P.-A. Fayolle, A. Pasko, E. Kartasheva, and N. Mirenkov. Shape recovery using functionally represented constructive models. In *Proceedings of SMI 2004 (In press)*, 2004.

[4] R. Fisher. Applying knowledge to reverse engineering problems. In *Proceedings of Geometric Modeling and Processing*, pages 149–155, 2002.

[5] J. H. Holland. *Adaptation in Natural and Artificial Systems*. 1975.

[6] C. Houck, J. Joines, and M. Kay. A genetic algorithm for function optimization: A matlab implementation. Technical report, NCSU-IE TR 95-09, 1995.

[7] S. Kirkpatrick, C. Gelatt, and M. Vecchi. Optimization by simulated annealing. *Science*, 220, 4598:671–680, 1983.

[8] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Verlag, 1996.

[9] J. Moré. The levenberg-marquardt algorithm implementation and theory. In *Lecture notes in mathematics No630 Numerical analysis*, volume 630, pages 105–116. New-York, 1978.

[10] Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. A multi-scale approach to 3d scattered data interpolation with compacly supported radial basis functions. In *Proceedings of Shape Modelling 2003*, 2003.

[11] Alexander Pasko, Valery Adzhiev, Alexei Sourin, and Vladimir Savchenko. Function representation in geometric modeling: concepts, implementation and applications. *The visual Computer*, 11:429–446, 1995.

[12] Jeffrey Smith. *Three Applications of Optimization in Computer Graphics*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, May 2003.

CONSTRUCTIVE TREE RECOVERY USING GENETIC ALGORITHMS

Pierre-Alain Fayolle
University of Aizu
Aizu-Wakamatsu, Japan
d8052103@u-aizu.ac.jp

Alexander Pasko
Hosei University
Tokyo, Japan
pasko@k.hosei.ac.jp

Nikolay Mirenkov
University of Aizu
Aizu-Wakamatsu, Japan
nikmir@u-aizu.ac.jp

Christophe Rosenberger
ENSI de Bourges - Université d'Orléans
Bourges, France
christophe.rosenberger@ensi-bourges.fr

Christian Toinard
ENSI de Bourges - Université d'Orléans
Bourges, France
christian.toinard@ensi-bourges.fr

ABSTRACT
An algorithm is described for recovering a constructive tree representation of a solid from a segmented point-set. The term point-set refers here to a finite set of points on or near the surface of the solid. Constructive geometry refers to the construction of complex solids by recursively applying set operations to simple primitives. It can be implemented on a computer by using a tree data structure with geometric primitives (planes, spheres and others) in the leaves and set-operations in the internal nodes. This tree data structure is called a constructive tree. A constructive tree can be syntactically translated into representations of solids by real-valued functions with the theory of R-functions. The recovered constructive tree is a correct representation of the point-set if the solid defined by the corresponding function matches the solid defined by the point-set. The search for a constructive tree is performed by a genetic algorithm. The formulation of the problem, the genetic algorithm and its parameters are discussed here.

KEY WORDS
Function Representation, R-functions, constuctive modeling, genetic algorithms, modeling automation.

1 Introduction

A problem of interest in computer aided design is the reverse-engineering of geometric models; in particular, we will consider here the problem of recovery of a constructive tree representation of a solid from a point-set.

A point-set associated to a given solid consists in a finite set of points lying on or near the surface of the solid. Such data can usually be obtained with devices like a laser scanner, a CT scanner or an MRI machine. We furthermore assume that the point-set is segmented, which means that the points within the point-set are clustered according to their belonging to some known primitives. Practically, we fit primitives from a dictionary of functions (quadric primitives) to do the clustering / fitting simultaneously.

Constructive geometry refers to the construction of complex solids by recursively applying set-operations (intersection, union and difference) to simple primitives. It can be implemented on a computer by using a tree data structure with geometric primitives in the leaves like for instance planes, spheres or others and set-operations between the primitives in the internal nodes. The point membership classification algorithm used to determine if a point belongs to a solid, can be implemented by a recursive tree traversal. Constructive Solid Geometry or CSG [1] restricts the operations to regularized set-theoretic operations and rigid body motions. A constructive tree can be trivially (syntactically) translated into representations of solids by real-valued functions with the theory of R-functions [2, 3]. The set of primitives can be extended with metaballs, convolution objects and others while the set of operations can be extended by other operations than the set-theoretic: deformations like tapering, offsetting, twisting, blending operations with added or subtracted material for example. The theory of Function Representation, FRep, describes it in details [4]. Constructive geometry helps to model complex shapes from simple "blocks" and is often used in computer aided design and implemented in most of the solid modelers.

Within the theory of R-functions and FRep, a solid is defined by the sign of a multivariate real-valued continuous function. The surface and the interior of a solid are defined by $\{x \in R^n : f(x) \geq 0\}$ and the exterior by: $\{x \in R^n : f(x) < 0\}$ where f is a real-valued continuous function. Representing a 2d solid, is done by the sign of a bivariate real-valued continuous function and a 3d solid by the sign of a trivariate real-valued continuous function.

Given a segmented point-set, we want to recover a constructive tree expressing the relations between the primitives. All the points from the point-set being on or near the surface of the solid, it means that the FRep associated to the given constructive tree takes 0 value on these points. The problem of constructive tree recovery is therefore equivalent to a combinatorial problem, where the search space is the finite set of all possible constructive trees that can be constructed with the given primitives. We will solve this problem with a genetic algorithm.

Previous Works Reverse engineering of geometric models refers usually to the following four-stage process: data

capture, preprocessing, segmentation and fitting of primitives and creation of the CAD model [5], where the CAD model usually refers to a Boundary Representation, or BRep, model. The work [5] presents a general introduction to the overall process of reverse engineering, with details on each of the pipeline's stages. More details on segmentation and fitting of primitives can be found in [6], see also the references therein. Segmentation / fitting is a difficult topic, still an abundant source of research and will not be discussed here, as well as the other two first stages of the reverse engineering's pipeline.

In all the works on reverse engineering, the final CAD model is available as a BRep model; we consider here the problem of creating a constructive model. Constructive modeling and Constructive Solid Geometry [1] are popular approaches in solid modeling and available in some of the CAD softwares. Furthermore, solid representation based on set operations and constructive solid geometry appear to solve technical problems inherent to the boundary representation. One of the possibility to create a constructive model is to follow the above pipeline and first generate a BRep model, then to apply algorithms for BRep to CSG conversion. Shapiro and Vossler proposed algorithms for BRep to CSG conversion in [7, 8, 9]; they also discuss the concept of separation: some boundary models can not be described by a CSG representation without additional half-spaces. Later an improved algorithm was proposed in [10].

In 2d, the problem of BRep to CSG conversion can be solved using the convex deficiency tree algorithm for linear and curved primitives (see [11, 12, 13]).

The main contribution of the paper is an algorithm that recovers a FRep defined by a constructive tree from a segmented point-set; The algorithm proposed here is based on a genetic algorithm: it is independent of the dimension and relatively simple to implement. It is general enough to use operations other than the set-theoretic operations: union, intersection, and differences. For example, blending union, and blending intersection can be used as valid operations.

The problem is formulated in Section 2 in terms of a minimization problem to be solved by a genetic algorithm; the genotype (encoding of the solutions), the choice of the genetic algorithm and its main parameters, as well as some other theoretical issues are discussed in Section 3. Section 4 proposes some results on mechanical parts taken from the real world. In Section 5, we further discuss the results and propose some extension and possible area for further works.

2 Description of the algorithm

We discuss now the algorithm used to recover an FRep model defined by a constructive tree using a segmented point-set and a list of fitted primitives. Note that the same algorithm can be applied for example to recover a FRep model from a boundary representation (BRep), since the BRep model naturally provides both the point-set and the primitives.

Suppose that we have a set of points $\{p_1, \ldots, p_n\}$ on or near the surface of the solid and a set of primitives $\{f_1, \ldots, f_m\}$ fitted to the segmented point-set. Given a finite set of possible operations between these primitives $\{\lambda_1, \ldots, \lambda_l\}$, we are searching for an ordering of the primitives with operations acting on them such that the formula:

$$F = f_{i_1} \lambda_{j_1} \ldots f_{i_m} \tag{1}$$

is a correct FRep model for the solid defined by the point-set. In the above expression, $j_k \in \{1, \ldots, l\}$ and the set $\{i_1, \ldots, i_m\}$ is obtained from the set $\{1, \ldots, m\}$ by a bijection and is used to order the primitives. A correct FRep model means that the defining function F satisfies $F > 0$ inside the solid, $F < 0$ outside the solid and $F = 0$ on the boundary of the solid, defined by the discrete set of points.

If this formula is evaluated from left to right, it is clear that it corresponds to a tree structure (left unbalanced) with operations in the internal nodes and primitives in the leaves. Evaluation from left to right, using an intermediate variable $temp$, is done as follow:

$$\begin{aligned} temp &\leftarrow f_{i_1}\ \lambda_{j_1}\ f_{i_2} \\ temp &\leftarrow temp\ \lambda_{j_2}\ f_{i_3} \\ &\ldots \\ temp &\leftarrow temp\ \lambda_{j_{m-1}}\ f_{i_m} \end{aligned} \tag{2}$$

This representation comes from the fact that we need to encode each FRep in an individual to be evaluated by the genetic algorithm. This representation suits well the encoding, and is easy to evaluate. The question is whether every constructive FRep can be encoded in that way. If the operations are only set-theoretic, then any formula can be represented by a left unbalanced representation. Using deMorgan transformations: $X \setminus Y \rightarrow X \cap \overline{Y}$, $\overline{X \cap Y} \rightarrow \overline{X} \cup \overline{Y}$, $\overline{X \cup Y} \rightarrow \overline{X} \cap \overline{Y}$, and $\overline{(\overline{X})} = X$, we transform the expression to an equivalent expression containing only $\cup$ and $\cap$. Then exploiting the fact that $\cup$ and $\cap$ are commutative, we can switch internal nodes of the formula to obtain a left unbalanced representation.

Using blending operations [14], union or intersection, the representation of a constructive FRep by a left unbalanced representation is still possible when the blend is symmetric; if the blend is not symmetric, then the operations are not commutative, and a left unbalanced representation may not exist. Unary operations like space deformations [15]: rotations, scaling or any other inverse space mappings, do not pose any problems at all and can be appended to the primitives. Actually these operations should be incorporated in the process of fitting the primitives to the segmented point-set.

Note that, using deMorgan laws to transform a constructive tree to a left heavy representation introduces the complement of point-sets. Practically it means that the primitives are oriented. When converting BRep to FRep it

may generally be the case, since the BRep model requires a global and consistent orientation. When fitting primitives to a point-set, the orientation can be obtained from the orientation of the point-set, which is a difficult problem [16].

Because the points $\{p_1, \ldots, p_n\}$ belong to the surface of the solid, the formula should evaluate to 0 at all of these points. With the notation, $F = f_{i_1}\lambda_{j_1} \ldots f_{i_m}$, it means $\forall p_i$, $F(p_i) = 0$. The problem can be reformulated as the search of a formula such that $\sum_{p_i} F(p_i)^2$ is minimum. A genetic algorithm is used for the minimization problem.

An individual of the population consists simply in an encoding of the left heavy representation of a FRep constructive tree. It is done by using an array of pairs. Each individual contains m pairs (op_k, L_k), $1 \leq k \leq m$, encoding the type of operations – op_k is an index to one of the operations from the set of possible operations – and L_k the position of the primitive k in the expression.

3 Implementation

A simple genetic algorithm, as introduced and discussed by Goldberg [17], is used. This genetic algorithm uses non-overlapping populations. For each generation, a new population is created by selecting from the previous population, and mating to produce the offsprings for the new population. These two steps are repeated until the termination criterion is met.

The fitnees function ϕ is defined by:

$$\phi(g) = \sum_{p_i \in S} F_g(p_i)^2 \tag{3}$$

where g is an individual in the current population; S is the point-set, with points on or near the surface of the solid; F_g is the FRep encoded in the individual g and corresponding to a left unbalanced tree (Eq. 1).

3.1 The genotype and the decoding

We use a one dimensional binary string as the data structure to encode an individual in the genetic algorithm. An individual corresponds to an array of pairs (op_k, L_k) where op_k is an index to one of the operations, and L_k is the priority level – or position – of the primitive k in the expression. Each pair is encoded as a sequence of bits (a gene).

We use an example to explain in more details how the decoding and encoding of the individuals are performed; this example consists of 16 primitives and the possible operations are union, intersection and blending intersection. Each pair (op_k, L_k) can then be encoded by $2 + 4 = 6$ bits. An operation index op_k can be encoded in 2 bits, since three operations are considered. The last 4 bits are used to encode the index of the k^{th} primitive (priority level) L_k in the expression construction. An expression is thus encoded by a string of size $6 * 16$=96 bits.

For a given binary string, we need also to be able to reconstruct the expression to evaluate it. This can be done with the following procedure:

- select the next pair (op_k, L_k) with the lower L_k value
- in case of several pair with same L_k value, take the one with the minimal index k in the initial list

This procedure is illustrated in the following example. Let a string be encoded by the following 16 pairs (op_k, L_k). Where $op_k \in \{0, 1, 2\}$, and $L_k \in \{0, .., 15\}$. $k = 1 : (0, 15)$, $k = 2 : (0, 1)$, $k = 3 : (1, 0)$, $k = 4 : (1, 2)$, $k = 5 : (1, 1)$, $k = 6 : (1, 0)$, ..., $k = 16 : (1, 7)$.

Following the procedure described above, we select the pairs in the following order: 3, 6, 2, 5, 4 ...1 and then the reconstructed expression becomes: $F = (op_3)f_3 op_6 f_6 op_2 f_2 op_5 f_5 op_4 f_4 ... op_1 f_1$ The first operation is in parenthesis because it is not taken into account in the evaluation and is here only for the symmetry of the expression. Finally each operation can be replaced. Let us suppose that $op_i == 0$ corresponds to union (|), $op_i == 1$ for intersection (&), and $op_i == 2$ for blending intersection (&&); we get the following expression: $F = f_3 \& f_6 | f_2 \& f_5 \ldots | f_1$.

We can now define genetic operations to be applied on the individuals.

3.2 The genetic operations

There are three operations that need to be defined for a genetic algorithm: the selection (selection of individuals in the previous population), the crossover (mating selected individuals to create new offspring), and the mutation (mutate a selected individual). We discuss in the following the algorithms used for their implementations.

3.2.1 Selection:

The selection is implemented by the roulette wheel algorithm. The idea of the roulette wheel is to choose randomly between all the individuals of the current population, with a higher probability to select an individual with a good fitness value.

3.2.2 Mutation:

Let us suppose, that there are n pairs and that a pair is coded using m bits. If an individual is selected for a mutation, then the mutation is done as follow:

1. choose randomly a position in the string
2. find the beginning of the m bit sequence it belongs to
3. flip all the m bits of the sequence

3.2.3 Crossover:

The following crossover is used:

1. choose randomly a position in the string of parent 1
2. find the beginning of the m bit sequence it belongs to
3. find the symmetric position in the parent 2
4. swap the subparts of the parents to generate the two children

4 Results and Examples

The input of the algorithm is a set of points scattered on the surface of the solid and the primitives associated to a segmentation of the point-set. Point-set segmentation and fitting of primitives to the different subsets of the segmented point-set are obtained by a brute force algorithm. A genetic algorithm is run on a list of primitives (cube, box, sphere, torus, ellipsoid), the best fitted primitive is selected and the corresponding points from the point-set are removed, then we loop to the first step until the point-set contains only a few points. The set of operations used in each of the examples contains: intersection, union, blending intersection and blending union.

4.1 Example 1

The first example consists of a point-set of 9530 points and 10 primitives. These primitives include planes, spheres and cylinders and were recovered using a brute force genetic algorithm as described above. In fact, a box was fitted instead of planes, but it was later segmented to planes to increase the number of primitives. It should be noticed that the use of a different segmentation algorithm may have produced directly such planes instead of a box.

We used the genetic algorithm and the genetic operations described above; we used a probability of 0.1 for the mutation and of 0.6 for the crossover. Figure 1 illustrates the convergence after 200 generations of an initial population of size 1000.

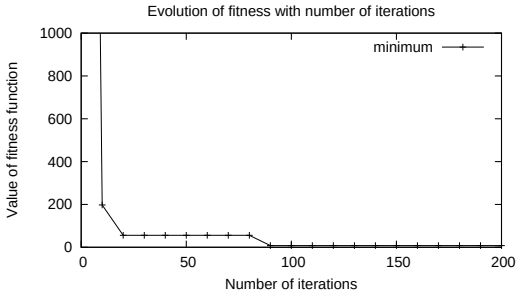


Figure 1. Evolution of the fitness of the best individual for a population of size 1000 for the first mechanical part

The solid resulting from the best individual is given in Fig. 2.

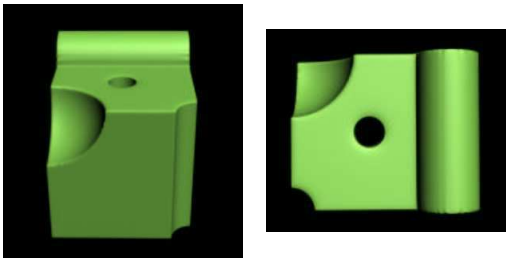


Figure 2. Result for the best individual for the first mechanical part

4.2 Example 2

The second example consists of a point-set of 49388 points segmented in 10 primitives. The same parameters as above are used for the size of population, probability of crossover and probability of mutation. Figure 3 illustrates the convergence after 200 generations of a population of size 1000. The solid obtained from the best individual is illustrated by Fig. 4.

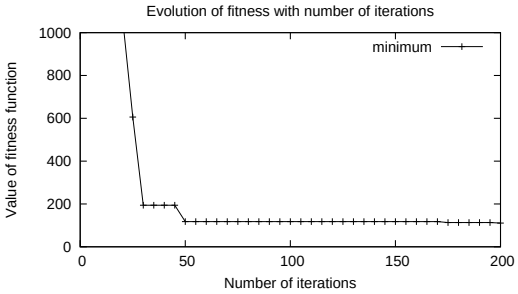


Figure 3. Evolution of the fitness of the best individual for a population of size 1000 for the second mechanical part

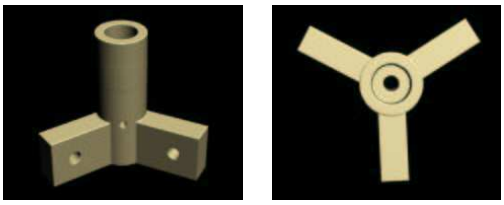


Figure 4. Result for the best individual for the second mechanical part

4.3 Comments

Comparing figures 1 and 3, we find a higher fitness value for the second example. It can be explained by a bigger number of points in the point-set used: the second point-set is approximately 5 times bigger. It can also be explained by a different accuracy in fitting the primitives. The number of points in the point-set has some influence on the algorithm. If this number is too big, it will slow down the evaluation of the fitness function and the overall speed of the algorithm.

5 Conclusion

We have presented a new algorithm based on a simple genetic algorithm to recover an FRep defined by a constructive tree from a segmented point-set and a list of fitted primitives. The algorithm is general, simple and can be easily extended to other operations than the set-theoretic ones: blending operation is an example. This algorithm exhibits similarities with genetic programming. The main difference is that genetic programming uses an infinite length representation for the solutions, where our problem has known finite length solutions. Still it may be interesting to investigate the behaviour of genetic programming and the found solutions on such problem.

Acknowledgements

P.-A. Fayolle acknowledges support by Monbukagakusho, the Japanese Ministry of Education, Culture, Sports, Science and Technology.
C. Toinard and C. Rosenberger would like to thank financial support provided by the Conseil Général du Cher.

References

[1] A. A. G. Requicha and H. B. Voelcker, Constructive solid geometry, Technical Report 25, Production Automation Project, University of Rochester, November 1977.

[2] V. L. Rvachev, Geometric applications of logic algebra, Technical report, Naukova Dumka, 1967, (in Russian).

[3] V. Shapiro, Theory of r-functions and applications: A primer, Technical Report TR91-1219, Computer Science Department, Cornell University, Ithaca, NY, 1991.

[4] A. Pasko, V. Adzhiev, A. Sourin, and V. Savchenko, Function representation in geometric modeling: Concepts, implementation and applications, *The Visual Computer*, 11(8), 1995, 429–446.

[5] T. Varady, R. R. Martin, and J. Cox, Reverse engineering of geometric models – an introduction, *Computer Aided Design*, 29(4), 1997, 255–268.

[6] P. Benko and T. Varady, Direct segmentation of smooth, multiple point regions, *Proc. of Geometric Modeling and Processing*, Tokyo, Japan, 2002, 169–178.

[7] V. Shapiro and D. L. Vossler, Construction and optimization of csg representations, *Computer-Aided Design*, 23(1), 1991, 4–20.

[8] V. Shapiro and D. L. Vossler, Efficient csg representation of two-dimensional solids, *Transaction of ASME, Journal of Mechanical Design*, 113, 1991, 292–305.

[9] V. Shapiro and D. L. Vossler, Separation for boundary to csg conversion, *ACM Transaction on Graphics*, 12(1), 1993, 35–55.

[10] S. Buchele and R. Crawford, Three-dimensional halfspace constructive solid geometry tree construction from implicit boundary representations, *Computer Aided Design*, 36(11), 2004, 1063–1073.

[11] V. L. Rvachev, L. V. Kurpa, N. G. Sklepus, and L. A. Uchishvili, Method of r-functions in problems on bending and vibrations of plates of complex shape, Technical report, Naukova Dumka, 1973 (in Russian).

[12] D. P. Peterson, Boundary to constructive solid geometry mappings: a focus on 3d issues, *Computer-Aided Design*, 18(1),1986, 3–14.

[13] V. Shapiro, A convex deficiency tree algorithm for curved polygons, *International Journal of Computational Geometry and Applications*, 11(2), 2001, 215–238.

[14] A. Pasko and V. Savchenko, Blending operations for the functionally based constructive geometry, *Proc. of CSG 94 Set-theoretic Solid Modeling: Techniques and Applications*, Winchester, UK, 1994, 151–161.

[15] V. Savchenko and A. Pasko, Transformation of functionally defined shapes by extended space mappings, *The Visual Computer*, 14(5), 1998, 257–270.

[16] H. Hoppe, *Surface reconstruction from unorganized points* (PhD thesis, Seattle, WA, USA, 1995).

[17] D. Goldberg, *Genetic Algorithms in search, optimization and machine learning* (Addison-Wesley, 1989).

Evolutionary Computation Approaches for Shape Modelling and Fitting

Sara Silva[1], Pierre-Alain Fayolle[2], Johann Vincent[3], Guillaume Pauron[3], Christophe Rosenberger[3], and Christian Toinard[4]

[1] Centro de Informática e Sistemas da Universidade de Coimbra, Polo II - Pinhal de Marrocos, 3030 Coimbra, Portugal
sara@dei.uc.pt

[2] University of Aizu, Software Department, AizuWakamatsu, Fukushima ken 965-8580, Japan
d8052103@u-aizu.ac.jp

[3] Laboratoire Vision et Robotique, UPRES EA 2078, ENSI de Bourges - Université d'Orléans, 10 boulevard Lahitolle, 18020 Bourges, France
{johann.vincent, guillaume.pauron, christophe.rosenberger}@ensi-bourges.fr

[4] Laboratoire d'informatique Fondamentale d'Orléans, CNRS FRE 2490, ENSI de Bourges - Université d'Orléans, 10 boulevard Lahitolle, 18020 Bourges, France
christian.toinard@ensi-bourges.fr

Abstract. This paper proposes and analyzes different evolutionary computation techniques for conjointly determining a model and its associated parameters. The context of 3D reconstruction of objects by a functional representation illustrates the ability of the proposed approaches to perform this task using real data, a set of 3D points on or near the surface of the real object. The final recovered model can then be used efficiently in further modelling, animation or analysis applications. The first approach is based on multiple genetic algorithms that find the correct model and parameters by successive approximations. The second approach is based on a standard strongly-typed implementation of genetic programming. This study shows radical differences between the results produced by each technique on a simple problem, and points toward future improvements to join the best features of both approaches.

1 Introduction

Shape modelling is a mature technology, extensively used in the industry for various applications (rapid prototyping, animation, modelling of cherubical prothesis, *etc*) [4]. Our purpose is to ease shape modelling of objects from the real world by fitting template shape models, defined by a functional representation (FRep) [17], to point data sets. The resulting model can later be modified and reused to fit the requirements of an application. An approach for modelling human body with template parameterized models was recently proposed [19]. Such a work underlines, in a different context, the importance to be able to later process and modify models from the real world.

The traditional methods used in reverse engineering and shape recovery of constructive solids rely on a segmentation of scan data and fitting of some mathematical models. Usually, these mathematical models are parametric or algebraic surface patches [3,7]. They are then converted to a boundary representation model. In [3,7], the need of relations between parameters or objects is introduced. These relations intend to guarantee symmetry or alignment in the object, thus enforcing the accuracy of the recovery procedure. Fitting parametric and algebraic surfaces, using relations between the parameters and objects, is a difficult problem of non-linear constrained optimization. Robertson *et al.* [18] proposes an evolutionary method based on GENOCOP III [14,15] to efficiently resolve this hard problem. The drawback of such methods in shape recovery is that they suit only boundary representation with segmented point sets. Adding new primitives would require a corresponding segmentation of the point sets, which is difficult or even impossible in the case of complex blends or sweeps. Furthermore, it may be difficult for the resulting model available only as a BRep (*i.e.* Boundary Representation) to be reused in extended modelling, analysis or animation.

We have extended [5] the general idea of knowledge-based recovery (*i.e.* the use of relations between parameters and primitives) with a different interpretation and a different model, the function representation of objects [17]. In this approach, standard shapes and relations are interpreted as primitives and operations of a constructive model. The input information provided by the user is a template (sketch) model, where the construction tree contains only specified operations and types of primitives while the parameters of operations and primitives are not required and are recovered by fitting. Template models may exist in dedicated library for each domain of applications, available to be reused, or else they need to be created by the user. In [5], a method based on a genetic algorithm is proposed for fitting the template FRep model to point sets. The main problem of this method is that the FRep model has to be defined by the user.

In this paper, we propose different Evolutionary Computation (EC) [1] approaches to automatically determine both the model (shape modelling) and its best parameters (shape fitting). We use both Genetic Algorithm (GA) [10,8] and Genetic Programming (GP) [12,2] methodologies, and discuss the results, pros and cons, and possible improvements of each approach.

2 The FRep model

In general, the shape recovery of objects follows a sequence of different steps, shown in Fig. 1. This paper is focused on the methods used to derive the FRep model and its parameters (modelling and model estimation).

2.1 The Function Representation

The function representation (FRep) [17] follows a constructive approach for modelling multidimensional objects. In the constructive modelling approach, a com-

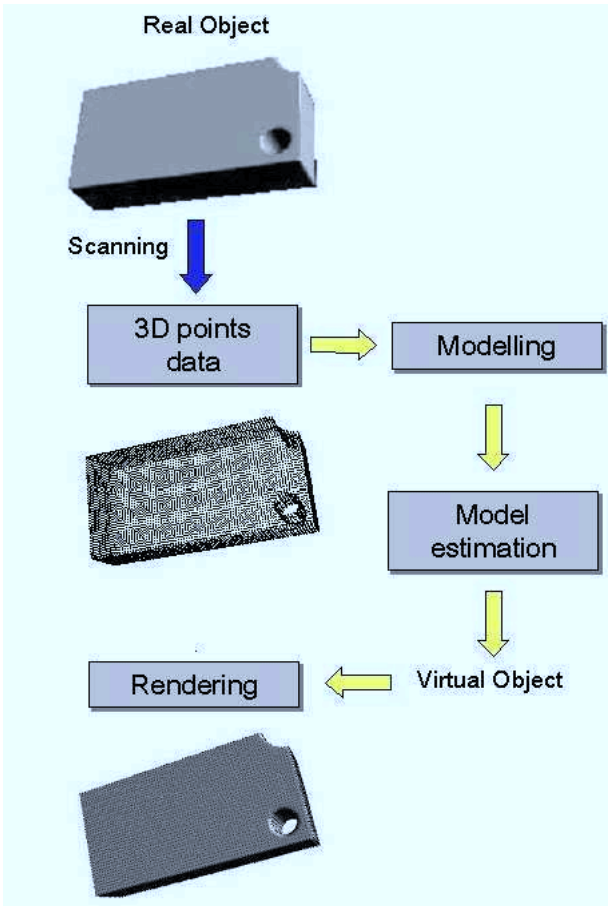


Fig. 1. Shape recovery of 3D objects.

plex object is decomposed into a set of simple ones, called primitives, that are then combined by different operations. The data structure describing the combination of primitives is called a construction tree. In the FRep model, the primitives are any continuous real valued functions, for example skeletal implicit surfaces, or algebraic surfaces. The operations themselves are defined analytically in such a manner that they yield continuous real valued functions as output, ensuring the closure property of the representation. Figure 2 represents the FRep model of the simple object illustrated in Fig. 1. Figure 3 represents the corresponding construction tree.

In the FRep model, a multidimensional object is defined by a single continuous real valued function f. The points x for which $f(x) \geq 0$ belong to the object, whereas the points for which $f(x) < 0$ are outside. In the construction tree, only

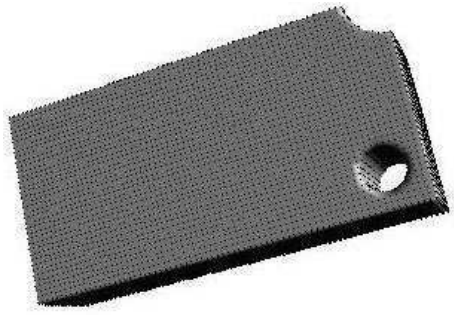


Fig. 2. Representation of the object in Fig. 1 by a FRep model

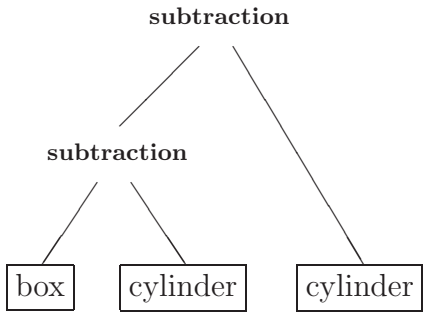


Fig. 3. Construction tree of the object shown in Fig. 2

specified operations and primitives are included, along with the parameters of each primitive, which must be tuned according to some modelling criteria.

2.2 Finding the Model

A FRep model classically uses 5 different primitives (sphere, box, cylinder in X, Y and Z) and 3 operations (intersection, union, subtraction). Even if we have a low number of primitives and operations, if we try to determine the FRep model of a simple object with 2 operators in its construction tree, we have $5 \times 3 \times 5 \times 3 \times 5 = 1125$ possible combinations. This is a very computationally expensive problem. This task is usually performed by a user but, depending on the object complexity, it may be too difficult to do. Once the model is determined, it is possible to find the best parameters [5] but, wanting to find an automatic approach to do both at the same time, we are dealing with an even more complex problem.

In the rest of the paper, the notation $f(\mathbf{x}, \mathbf{a})$ is used for a parameterized FRep, where $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$ is a point in 3D space and $\mathbf{a} = (a_1, \ldots, a_m) \in \mathbb{R}^m$ is a vector of m parameters.

2.3 Tuning the Parameters

Tuning the parameters is based on a set of 3D points, $S = \{\mathbf{x_1}, \ldots, \mathbf{x_N}\}$, scattered on the surface of the object. Given S, the task is to find the best configuration for the set of parameters $\mathbf{a}^* = (a_1, \ldots, a_m)$ so that the parameterized FRep model $f(\mathbf{x}, \mathbf{a}^*)$ closely fits the data points. The FRep model $f(\mathbf{x}, \mathbf{a})$, built in a constructive way, approximates the shape of the solid being reverse engineered. The vector of parameters $\mathbf{a}$ controls the final location and shape of the solid and thus the best fitted parameters should give the closest possible model according to the information provided by S.

The sphere primitive requires 4 parameters (a 3D point indicating the center, plus the radius), while the box primitive requires 5 parameters (3D center point, width, height and depth). Each cylinder primitive requires only 3 parameters (because each cylinder is infinite in its respective direction, a 2D center plus the radius is enough to completely define it).

2.4 Evaluating the Parameterized Model

In order to evaluate the mismatch between the discrete representation of the solid, given by S, and the surface of the solid for the current vector of parameters, implicitly defined by f, a fitness function is needed. The function f defines an algebraic distance between the current point of evaluation $\mathbf{x}$ and the surface it implicitly defines [17]. The fitness error thus becomes the square of the defining function values at all points of S (the surface of the solid being the set of points with zero function value):

$$error(\mathbf{a}) = \frac{1}{2} \sum_{i=1}^{N} f^2(\mathbf{x_i}, \mathbf{a}) \tag{1}$$

Our goal is to search for both the function f and the vector of parameters $\mathbf{a}^*$ minimizing (1). This is the fitness function used by the different approaches described next.

3 Evolutionary Computation Approaches

Given the characteristics of the FRep model described in the previous section, we realize that our goal is just a particular case of the general problem of evolving both a structure and its parameters at the same time. Different Evolutionary Computation (EC) approaches can be tried in order to find an efficient and reliable method to solve this task.

3.1 Genetic Algorithms within Genetic Programming

GP is known to be very good at evolving variable-length structures, while GAs are most appropriate for optimizing a fixed-length set of parameters. The first

idea to tackle the problem of doing both at the same time is very obvious: use a GP to evolve the construction tree, where the evaluation of each individual is preceded by the optimization of its parameters performed by a GA.

However, we have abandoned this approach for being too computationally demanding. A single run of standard GP evolving 500 individuals during 50 generations would launch a GA as many as $50 \times 500 = 25000$ times! Although this could be the ultimate and most appropriate solution for the problem, we quickly moved to other less demanding techniques.

3.2 Multiple Genetic Algorithms

Using a single GA to evolve both the construction tree and its parameters was not a promising option. Due to its typical fixed-length representation, the usage of a single GA would require the prior information regarding how many levels the construction tree should have. It would also require distinct genes for the structure and for the parameters, as well as the consequent additional care in terms of the genetic operators used in the evolution.

So, we have decided to try an automatic approach where multiple GAs are used to iteratively determine each level of the FRep construction tree. We try by this approach to reproduce the methodology of a sculptor for a real object. At each step, the object is refined until the desired level of precision. In our approach, at the first level, we determine the geometric primitive that best approximates the 3D points by launching 5 GAs, one for each primitive. Each GA only has to optimize the fixed set of parameters of the corresponding primitive by minimizing the fitness function (1) where f is already determined by the primitive. The GA achieving the lowest fitness determines the primitive to be used in the first level.

To determine the second level, we assume the primitive defined previously is correct and try to determine the operation and the other primitive. For this we must launch 15 GAs, one for each possible combination (3 operations $\times$ 5 primitives). These new GAs optimize the parameters of both the first and second primitives. An option could be made to always use the parameters found in the previous level, but recalculating them is more correct in terms of optimization.

After determining the best operation and primitive for the second level of the construction tree, 15 GAs are once again launched to determine the third level, and so on until the fitness reaches the minimum value of zero, or the user decides that the approximation is sufficiently good. The task of the GAs becomes more complex as more levels are determined. The results obtained with this approach are described in Sect. 4.1.

3.3 Strongly-Typed Genetic Programming

Using only GP to automatically evolve the construction tree and its parameters at the same time appeared to be a viable and promising option from the very beginning. GP typically uses variable-length representations, and its enormous flexibility make it easily adaptable to almost any type of problem.

However, standard GP [12] is not particularly suited for evolving structures where the closure property is not present. This is the case in our problem where, for instance, the only valid arguments of an operation are primitives or other operations, and the only valid arguments of a primitive are the parameters. So, instead of standard GP, we have used strongly-typed GP [16], a much suitable option for this type of problem.

We have defined two types in our strongly-typed system. The first type includes all the primitives (sphere, box, cylinder in X, Y and Z) and operations (intersection, union, subtraction) mentioned earlier. The second type refers to the parameters, and includes terminals and simple arithmetic functions. Elements of the first type can be further differentiated in two subtypes, because primitives can only accept parameters as input arguments, while operations can only accept primitives or other operations.

Two standard genetic operators are used in our system: tree crossover and tree mutation. We do not use different genetic operators for each type, but we do ensure that both crossover and mutation are type-aware, meaning that any two parents always create type-valid offspring. In crossover, the crossing point in the first tree is chosen randomly, and then the crossing point in the second tree is chosen such that swapping the subtrees creates a type-valid individual. In mutation, the crossing point is also chosen randomly, and the new replacing subtree is created such that the mutated individual is valid. A description of the results obtained with the strongly-typed GP approach can be found in Sect. 4.2.

4 Experimental Results

In all our experiments, we have tested the different approaches with the object used in figures 1 through 3, modelled for testing purposes using HyperFun [9], a set of tools for FRep geometric modelling. The surface of the object has been sampled to create a data set of 10714 3D points, represented in Fig. 4. This data set is used for calculating the fitness function (1) described in Sect. 2.4.

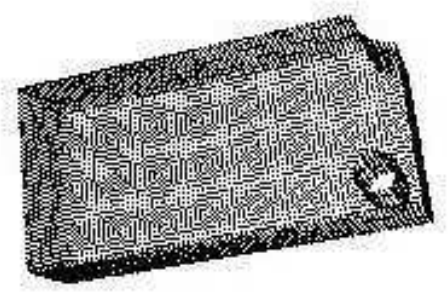


Fig. 4. Data set of 3D points used for the virtual modelling of the object

The FRep defining function F shown below has been determined as a parameterized model for the recovery process:

$$F(\mathbf{x}, \mathbf{a}) := subtraction(subtraction(box(\mathbf{x}, \mathbf{a}), cylinderZ(\mathbf{x}, \mathbf{a})), cylinderZ(\mathbf{x}, \mathbf{a})); \quad (2)$$

This FRep model consists of three simple primitives: one box and two infinite cylinders oriented along the Z axis, each primitive defined by its parameterized model. For example, in the case of the cylinderZ, the defining function is: $cylinder(x, a) := a[1]^2 - (x[1] - a[2])^2 - (x[2] - a[3])^2$, where $a[1]$, $a[2]$, and $a[3]$ are parameters meaning the radius, and the center (X,Y) of the cylinder, respectively. All the primitives are combined together using the subtraction operator (\), which is itself defined analytically as discussed in [17] (see Fig. 3 for the associated constructive tree).

4.1 Multiple Genetic Algorithms

The GA system used was the GAOT [11], implemented in ©MATLAB [22], with the main running parameters indicated in Table 1. Unspecified parameters were used with the default values.

Table 1. Main running parameters of the GA system used

Chromosome Type	real-valued
Population Size	1000
Mutation	non-uniform, probability 0.8
Crossover	arithmetic, 20 tries
Selection for Reproduction	normalized geometric ranking
Stop Criteria	100 generations

Table 2 shows the values of the fitness function achieved for each possible primitive at the first level of the construction tree. The lowest (best) fitness value was achieved for the box primitive, clearly the one that best fits the point data set. This successfully concludes the first iteration of the multiple GA approach.

Table 2. Value of the fitness function at the first level of the construction tree for each possible primitive

Sphere	CylinderZ	CylinderY	CylinderX	Box
717449.8	47747.2	749123.7	756618.1	**534.1**

Table 3 shows the values of the fitness function achieved for each possible combination of operation and primitive at the second level of the construction

tree. The lowest value corresponds to the cylinderZ and subtraction pair, which can be verified in Fig. 3 to be correct. Although the best fitness is not even close to zero, the structural elements chosen for the construction tree are correct. This means that, after finishing this iterative process, a single GA can be used to perform the full optimization of all the parameters of the structure, thus achieving high quality shape fitting.

Table 3. Value of the fitness function at the second level of the construction tree for each possible combination of primitive and operation

	Sphere	CylinderZ	CylinderY	CylinderX	Box
Intersection	230247.1	24756.7	712364.5	1502950.5	775.3
Union	749166.7	46059.7	31695.3	10156.8	78192.8
Substraction	176985.1	**544.2**	1827.9	1493.0	3463.3

At each subsequent iteration we adopt the same methodology, comparing 15 values of the fitness function derived by the multiple GAs. We always choose the operation and primitive pair that achieves the lowest fitness value.

4.2 Strongly-Typed Genetic Programming

The strongly-typed GP system used was an adaptation of GPLAB [20], a GP toolbox for ©MATLAB [22], with the main running parameters indicated in Table 4. Unspecified parameters were used with the default values.

Table 4. Main running parameters of the GP system used.

Function Set	$\{intersection, union, subtraction\}$ (type 1, subtype 1) $\{sphere, box, cylinderX \dots Z\}$ (type 1, subtype 2) $\{+, -\}$ (type 2)
Terminal Set	$\{rand, 0, 1\}$ (type 2)
Population Initialization	Ramped Half-and-Half [12]
Population Size	500
Maximum Tree Depth	initial: 3-6, final: 6-10 [21]
Operator Rates	crossover/mutation: 0.5/0.5
Reproduction Rate	0.1
Selection for Reproduction	tournament [13], size 2-50
Selection for Survival	replacement (no elitism)
Stop Criteria	100 generations

The function set contains all the elements indicated in the three sets of the table. Their types are described in Sect. 3.3. *rand* is a random constant between

0 and 1, created once and then used like the other constants. We have adopted some parsimony pressure measures in our GP system, namely the Lexicographic Parsimony Pressure tournament [13] and the Heavy Dynamic Limit [21] on tree depth, along with the traditional static limit [12]. We have used a range of values for the maximum tree depth and tournament size in different GP runs.

Figure 5(left plot) shows the results of a typical GP run obtained with a tournament of size 50 (10% of the population) and an initial tree depth of 6 with maximum value 10. It is immediately apparent that GP is able to easily achieve good fitness values, and keep improving them as the evolution proceeds (best value of this run was 20.7). However, the typical GP solution is not parsimonious at all, obtaining good approximations only thanks to an extensive and creative combination of primitives, one that does not reflect the simplicity of the object being modelled. For instance, the run illustrated in the left plot produces a solution containing 46 operations, 47 primitives, and 76+218 parameter elements (arithmetic constants and constants), totalling 387 nodes in the construction tree. The first primitive used in this tree was not the box.

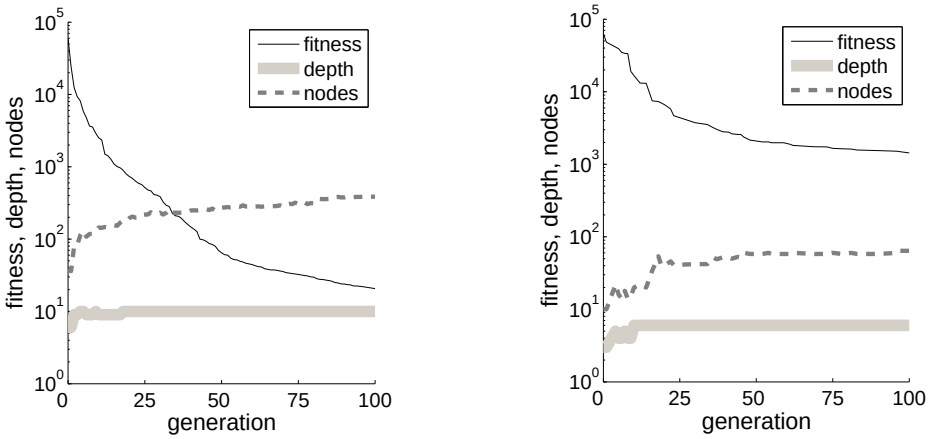


Fig. 5. Results of typical GP runs, with high selective pressure and low parsimony pressure *(left)*, and with low selective pressure and high parsimony pressure *(right)*

On the attempt to produce shorter and more realistic solutions, we have increased the parsimony pressure by using an initial tree depth of only 3, with maximum value 6. We have also decreased the selective pressure to allow a larger exploration of the search space, by reducing the tournament size to 2. Figure 5 (right plot) illustrates a typical run obtained with these parameters. The convergence to better fitness values was much more difficult (best value of this run was 1434.4), and the solution produced was much smaller, but still far from expected, containing 8 operations, 9 primitives, and $10 + 37$ parameter elements (operators and constants), totalling 64 nodes in the construction tree. Once again, the first primitive used was not the box.

5 Conclusions and Future Work

We have extended the work presented in [6] for determining and fitting a parameterized FRep model to a 3D point data set. We have proposed different EC approaches and shown that they produce radically different results when applied to a simple problem of shape modelling and fitting.

The multiple GA approach produces the correct construction tree, simple and compact as a human user would do. The process can be performed in two phases, first iteratively deriving the tree and later fine tuning its parameters with a final single GA. The disadvantage of this approach is that launching so many GAs is computationally expensive, especially when we consider that each level of the construction tree demands more from each GA. Another problem with the approach is its user dependence, at least in its current implementation where the system is not autonomous enough to decide when to add more levels to the construction tree, and when to stop. However, this can also be regarded as an advantage, because no one better than the user can decide which level of detail is necessary for the current application.

The strongly-typed GP approach is able to produce highly fit solutions in a totally automatic manner. However, the construction trees generated suffer from the extreme lack of parsimony that usually plagues GP evolution, even when parsimony pressure measures are applied. The creative solutions produced by GP do not please the practitioners of the field. Further work should be performed in order to make GP operate more like the iterative generation of the construction trees. Starting from very small trees and regular genetic operators to optimize the parameters, other genetic operators could be used to specifically add primitives to the tree, producing a similar behavior to the multiple GA approach.

Real-world objects of higher complexity should be used to test both approaches, reflecting the more realistic conditions where the shape modelling and fitting problem is too difficult to be performed by the user. Under such conditions, the multiple GA approach may not be able to produce such simple and clear-cut solutions in a feasible amount of time, and the creativeness and flexibility of the GP approach may become essential in producing acceptable results.

Regardless of what we may expect from future results, we believe that somewhere among the proposed approaches are already the necessary ingredients to achieve a fully automatic EC solution for the shape modelling and fitting problem, hopefully readily applicable to other problems dealing with the same issue of optimizing a model and its parameters at the same time.

Acknowledgements

This work was partially financed by grant SFRH/BD/14167/2003 from Fundação para a Ciência e a Tecnologia, Portugal. We would like to thank the members of the ECOS – Evolutionary and Complex Systems Group (Universidade de Coimbra), in particular group leader Ernesto Costa, for the valuable ideas and suggestions provided regarding this work.

References

1. Back, T., Fogel, D.B., Michalewicz, Z. (eds.): Handbook of Evolutionary Computation. IOP Publishing and Oxford University Press, New York and Bristol (1997)
2. Banzhaf, W., Nordin, P., Keller, R.E., Francone, F.D.: Genetic Programming - An Introduction. Morgan Kaufmann, San Francisco (1998)
3. Benko, P., Kos, G., Varady, T., Andor, L., Martin, R.: Constrained Fitting in Reverse Engineering. Computer Aided Geometric Design **19** (2002) 173–205
4. Costantini, F., Toinard, C.: Collaboration and Virtual Early Prototyping Using the Distributed Building Site Metaphor. In: Rahman, S.M.M. (ed.): Multimedia Networking: Technology, Management and Applications (2002) 290–332
5. Fayolle, P.-A., Rosenberger, C., Toinard, C.: Shape Recovery and Functional Modeling Using Genetic Algorithms. In: Proceedings of IEEE LAVAL VIRTUAL (2004) 227–232
6. Fayolle, P.-A., Pasko, A., Kartasheva, E., Mirenkov, N.: Shape Recovery Using Functionally Represented Constructive Models. In: Proceedings of SMI 2004 (2004) (in press)
7. Fisher, R.: Applying Knowledge to Reverse Engineering Problems. In: Proceedings of Geometric Modeling and Processing. IEEE Computer Society (2002) 149–155
8. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Boston, MA (1989)
9. HyperFun project (2005) `http://cis.k.hosei.ac.jp/~F-rep/HF_proj.html`
10. Holland, J.H.: Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor (1975)
11. Houck, C., Joines, J., Kay, M.: A Genetic Algorithm for Function Optimization: A Matlab Implementation. Technical Report NCSU-IE TR 95-09 (1995)
12. Koza, J.R.: Genetic Programming –On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge, MA (1992)
13. Luke, S., Panait, L.: Lexicographic Parsimony Pressure. In: Langdon, W.B. *et al.* (eds.): Proceedings of GECCO-2002. Morgan Kaufmann (2002) 829–836
14. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag, Berlin (1996)
15. Michalewicz, Z., Fogel, D.B.: How to Solve It: Modern Heuristics, 2nd edn. Springer, Berlin (2004)
16. Montana, D.J.: Strongly Typed Genetic Programming. BBN Technical Report #7866 (1994)
17. Pasko, A., Adzhiev, V., Sourin, A., Savchenko, V.: Function Representation in Geometric Modeling: Concepts, Implementation and Applications. The Visual Computer **11** (1995) 429–446
18. Robertson, C., Fisher, R., Werghi, N., Ashbrook, A.: An Evolutionary Approach to Fitting Constrained Degenerate Second Order Surfaces. EvoWorkshops (1999) 1–16.
19. Seo, H., Magnenat-Thalmann, N.: An Example-Based Approach to Human Body Manipulation. Graphical Models **66** (2004) 1–23
20. Silva, S.: GPLAB – A Genetic Programming Toolbox for MATLAB (2005) `http://gplab.sourceforge.net`
21. Silva, S., Costa, E. Dynamic Limits for Bloat Control - Variations on Size and Depth. In: Deb, K., Poli, R., Banzhaf, W. *et al.* (eds.): Proceedings of GECCO-2004. Springer (2004) 666–677
22. The MathWorks – MATLAB and Simulink for Technical Computing (2005) `http://www.mathworks.com/`

Shape Modeling
With Genetic Programming

P.-A. Fayolle[1], S.Silva[2], G. Latinier[3], D. Saffrey[3], C. Rosenberger[4], C. Toinard[5]

[1] University of Aizu, Software Department
AizuWakamatsu, Fukushima ken 965-8580, Japan
email: d8052103@u-aizu.ac.jp

[2] Centro de Informática e Sistemas da Universidade de Coimbra,
Polo II - Pinhal de Marrocos, 3030 Coimbra, Portugal
email:sara@dei.uc.pt

[3] ENSI de Bourges, 10 boulevard Lahitolle, 18020 Bourges, France
email:gregory.latinier,david.saffrey@ensi-bourges.fr

[4] Laboratoire Vision et Robotique, UPRES EA 2078
10 boulevard Lahitolle, 18020 Bourges, France
email:christophe.rosenberger@ensi-bourges.fr

[5] Laboratoire d'informatique Fondamentale d'Orléans, CNRS FRE 2490
10 boulevard Lahitolle, 18020 Bourges, France
email: christian.toinard@ensi-bourges.fr

Abstract

The paper proposes a method extending [5] for shape recovery of 3D models given a dataset of 3D points on or near the surface of a real object. The developed method is based on genetic programming and provides at the same time the modeling and fitting of a 3D object. The parametrized shapes are defined by the so-called function representation (FRep in short)[17]. The final recovered model (available as a FRep) can then be used efficiently in further modeling, animation or analysis. The main advantage of this representation is its low storage cost. We illustrate in the paper some preliminary modeling results of 3D objects.

Keywords: *shape recovery, rapid prototyping, genetic programming, function representation, 3D reconstruction.*

1 Introduction

Shape modelling is a mature technology, extensively used in the industry for various applications (rapid prototyping, animation, modelling of cherubical prothesis, *etc*) [4]. Let us give a short example for the rapid prototyping of an aircraft [4]. The design of a cabin starts from the scanning of a real aircraft seat. Then, the designer adjusts the different parts (e.g. arms, back) of that generic seat according to the requirements of the cabin. As shown by this example, a realistic rendering of the real seat is not enough, it is necessary to handle the generic seat model in terms of different parts representing not only solids but also functional components. Later the model can be extended by a designer with extra components.

Our purpose is to ease shape modeling of objects from the real world by fitting template shape models, defined by a functional representation (FRep) [17], to point data sets. The resulting model can later be modified

and reused to fit the requirements of an application. An approach for modeling human body with template parameterized models was recently proposed [19]. Such a work underlines, in a different context, the importance to be able to later process and modify models from the real world.

The traditional methods used in reverse engineering and shape recovery of constructive solids rely on a segmentation of scan data and fitting of some mathematical models. Usually, these mathematical models are parametric or algebraic surface patches [3, 7]. They are then converted to a boundary representation model. In [3, 7], the need of relations between parameters or objects is introduced. These relations intend to guarantee symmetry or alignment in the object, thus enforcing the accuracy of the recovery procedure. Fitting parametric and algebraic surfaces, using relations between the parameters and objects, is a difficult problem of non-linear constrained optimization. Robertson *et al.* [18] proposes an evolutionary method based on GENOCOP III [14, 15] to efficiently resolve this hard problem. The drawback of such methods in shape recovery is that they suit only boundary representation with segmented point sets. Adding new primitives would require a corresponding segmentation of the point sets, which is difficult or even impossible in the case of complex blends or sweeps. Furthermore, it may be difficult for the resulting model available only as a BRep (*i.e.* Boundary Representation) to be reused in extended modelling, analysis or animation.

We have extended [5] the general idea of knowledge-based recovery (*i.e.* the use of relations between parameters and primitives) with a different interpretation and a different model, the function representation of objects [17]. In this approach, standard shapes and relations are interpreted as primitives and operations of a constructive model. The input information provided by the user is a template (sketch) model, where the construction tree contains only specified operations and types of primitives while the parameters of operations and primitives are not required and are recovered by fitting. Template models may exist in dedicated library for each domain of applications, available to be reused, or else they need to be created by the user. In [5], a method based on a genetic algorithm is proposed for fitting the template FRep model to point sets. The main problem of this method is that the FRep model has to be defined by the user.

In this paper, we propose a new approach to automatically determine both the model (shape modelling) and its best parameters (shape fitting). In order to achieve this goal, Genetic Programming (GP) [12, 2] is used. We show some experimental results and we conclude on perspectives of this study.

2 Developed Method

In general, the shape recovery of objects follows different steps (see figure 1). This paper deals with shape recovery of parametrized function representation (FRep). We focus indeed in this paper on the method used to determine the model and estimate the associated parameters that extends the approach of [5] by using genetic programming.

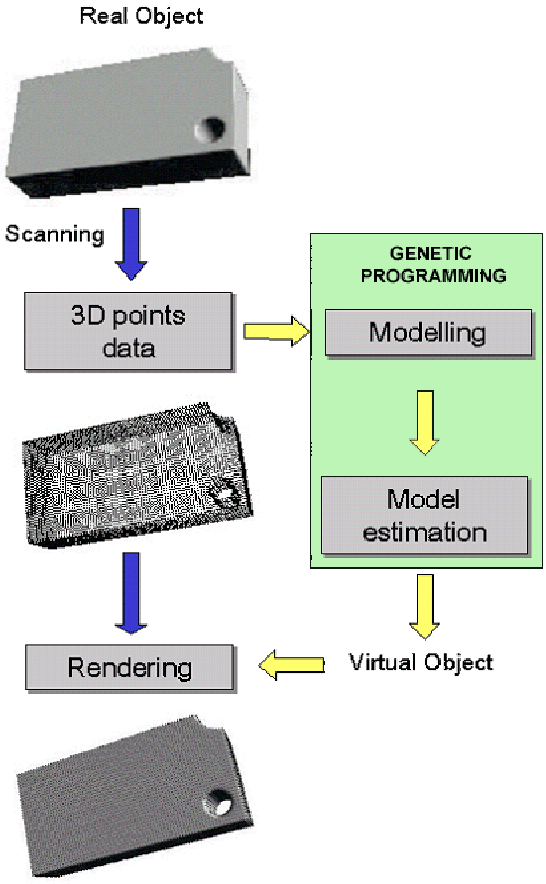


Figure 1. 3D reconstruction of shapes.

The function representation (FRep) [17] follows a constructive approach for modelling multidimensional objects. In the constructive modelling approach, a complex object is decomposed into a set of simple ones, called primitives, that are then combined by different operations. The data structure describing the combination of primitives is called a construction tree. In the FRep model, the primitives are any continuous real valued functions, for example skeletal implicit surfaces, or algebraic surfaces. The operations themselves are defined analytically in such a manner that they yield continuous real valued functions as output, ensuring the closure property of the representation. Figure

2 represents a 3D simple object and figure 3 shows its corresponding construction tree.

Figure 2. Example of an object to model

In the FRep model, a multidimensional object is defined by a single continuous real valued function f. The points x for which $f(x) \geq 0$ belong to the object, whereas the points for which $f(x) < 0$ are outside. In the construction tree, only specified operations and primitives are included, along with the parameters of each primitive, which must be tuned according to some modelling criteria.

2.1 Finding the Model

A FRep model classically uses 5 different primitives (sphere, box, cylinder in X, Y and Z) and 3 operations (intersection, union, subtraction). We restrict the proposed method to paraxial primitive objects. The methodology would be identical by adding rotations operators (but with a higher number of parameters to estimate).

Even if we have a low number of primitives and operations, if we try to determine the FRep model of a simple object with 2 operators in its construction tree, we have $5 \times 3 \times 5 \times 3 \times 5 = 1125$ possible combinations. This is a very computationally expensive problem. This task is usually performed by a user but, depending on the object

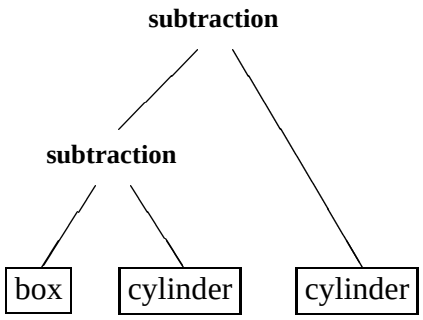


Figure 3. Construction tree of the object shown in Fig. 2

complexity, it can be very difficult to do. Once the model is determined, it is possible to find the best parameters [5] but, wanting to find an automatic approach to do both at the same time, we are dealing with an even more complex problem.

In the rest of the paper, the notation $f(\mathbf{x}, \mathbf{a})$ is used for a parameterized FRep, where $\mathbf{x} = (x, y, z) \in \mathbb{R}^3$ is a point in 3D space and $\mathbf{a} = (a_1, \ldots, a_m) \in \mathbb{R}^m$ is a vector of m parameters.

2.2 Tuning the Parameters

Tuning the parameters is based on a set of 3D points, $S = \{\mathbf{x_1}, \ldots, \mathbf{x_N}\}$, scattered on the surface of the object. Given S, the task is to find the best configuration for the set of parameters $\mathbf{a}^* = (a_1, \ldots, a_m)$ so that the parameterized FRep model $f(\mathbf{x}, \mathbf{a}^*)$ closely fits the data points. The FRep model $f(\mathbf{x}, \mathbf{a})$, built in a constructive way, approximates the shape of the solid being reverse engineered. The vector of parameters $\mathbf{a}$ controls the final location and shape of the solid and thus the best fitted parameters should give the closest possible model according to the information provided by S.

The sphere primitive requires 4 parameters (a 3D point indicating the center, plus the radius), while the box primitive requires 5 parameters (3D center point, width, height and depth). Each cylinder primitive requires only 3 parameters (because each cylinder is infinite in its respective direction, a 2D center plus the radius is enough to completely define it).

2.3 Evaluating the Parameterized Model

In order to evaluate the mismatch between the discrete representation of the solid, given by S, and the surface of the solid for the current vector of parameters, implicitly defined by f, a fitness function is needed. The function f defines an algebraic distance between the current point of evaluation $\mathbf{x}$ and the surface it implicitly defines [17]. The fitness error thus becomes the square of the defining function values at all points of S (the surface of the solid being the set of points with zero function value):

$$error(\mathbf{a}) = \sum_{i=1}^{N} f^2(\mathbf{x_i}, \mathbf{a}) \qquad (1)$$

Our goal is to search for both the function f and the vector of parameters $\mathbf{a}^*$ minimizing equation (1). This is the fitness function used by the different approaches described next.

2.4 Strongly-Typed Genetic Programming

Using only GP to automatically evolve the construction tree and its parameters at the same time appeared to be a viable and promising option from the very beginning. GP typically uses variable-length representations, and its enormous flexibility make it easily adaptable to almost any type of problem.

However, standard GP [12] is not particularly suited for evolving structures where the closure property is not present. This is the case in our problem where, for instance, the only valid arguments of an operation are primitives or other operations, and the only valid arguments of a primitive are the parameters. So, instead of standard GP, we have used strongly-typed GP [16], a much suitable option for this type of problem.

We have defined two types in our strongly-typed system. The first type includes all the primitives (sphere, box, cylinder in X, Y and Z) and operations (intersection, union, subtraction) mentioned earlier. The second type refers to the parameters, and includes terminals and simple arithmetic functions. Elements of the first type can be further differentiated in two subtypes, because primitives can only accept parameters as input arguments, while operations can only accept primitives or other operations.

Two standard genetic operators are used in our system: tree crossover and tree mutation [22]. We do not use different genetic operators for each type, but we do ensure that both crossover and mutation are type-aware, meaning that any two parents always create type-valid offspring. In crossover, the crossing point in the first tree is chosen randomly, and then the crossing point in the second tree is chosen such that swapping the subtrees creates a type-valid individual. In mutation, the crossing point is also chosen randomly, and the new replacing subtree is created such that the mutated individual is valid.

2.5 Visualisation of FREP models

A generated FREP model is evaluated by considering the value of the fitness function defined by equation (1). The visualisation of the FREP model is also used to check its shape. Figure 4 present 4 FREP models.

3 Experimental results

The strongly-typed GP system used was an adaptation of GPLAB [20], a GP toolbox for ©MATLAB [22], with the main running parameters indicated in Table 1. Unspecified parameters were used with the default values.

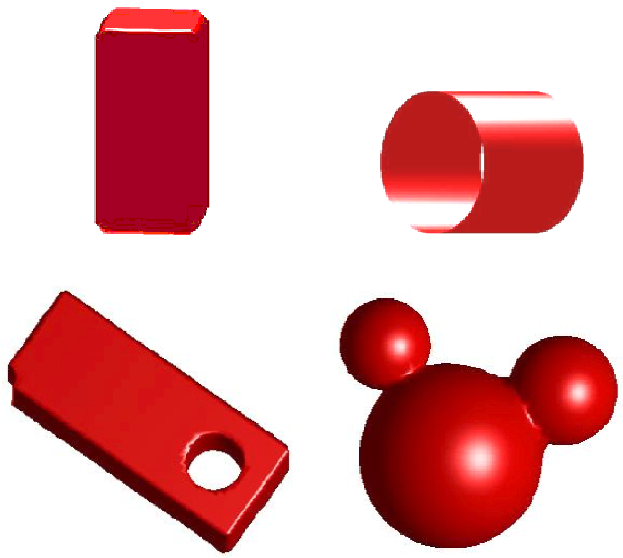


Figure 4. Some example of FREP models

Function Set	$\{intersection, union, subtraction\}$ $\{sphere, box, cylinder X \dots Z\}$ $\{+, -\}$
Terminal Set	$\{rand, 0, 10\}$
Population Initialization	Ramped Half-and-Half [12]
Population Size	500
Maximum Tree Depth	initial: 3-6, final: 6-10 [21]
Operator Rates	crossover/mutation: 0.5/0.5
Reproduction Rate	0.1
Selection for Reproduction	tournament [13], size 2-50
Selection for Survival	replacement (no elitism)
Stoping Criterion	100 generations

Table 1. Main running parameters of the GP system used.

The function set contains all the elements indicated in the three sets of the table. Their types are described in Sect. 2.4. $rand$ is a random constant between 0 and 10, created once and then used like the other constants. We have adopted some parsimony pressure measures in our GP system, namely the Lexicographic Parsimony Pressure tournament [13] and the Heavy Dynamic Limit [21] on tree depth, along with the traditional static limit [12]. We have used a range of values for the maximum tree depth and tournament size in different GP runs.

We present in figure 5 the visualisation of a FREP model and the generated tree. The generated FREP model can appear visually very simple and characterized by a low value of the fitness function (fitness value less than 10^{-5}). The associated tree can be very complex such as the one in figure 5.

In the following experiments, we have tested the proposed approach with the object used in figure 2, modeled for testing purposes using HyperFun [9], a set of tools for FRep geometric modelling. The surface of the object has been sampled to create a data set of 10714 3D points, represented in Fig. 6. This data set is used for calculating the fitness function (1) described in Sect. 2.3.

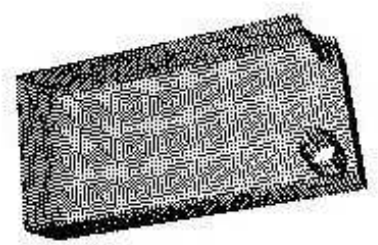


Figure 6. Data set of 3D points used for the virtual modelling of the object

The FRep defining function F shown below has been determined as a parameterized model for the recovery process:

$$F(\mathbf{x},\mathbf{a}) := subtraction(subtraction(box(\mathbf{x},\mathbf{a}), cylinderZ(\mathbf{x},\mathbf{a})), cylinderZ(\mathbf{x},\mathbf{a})); \quad (2)$$

This FRep model consists of three simple primitives: one box and two infinite cylinders oriented along the Z axis, each primitive defined by its parameterized model. For example, in the case of the cylinderZ, the defining function is: $cylinder(x,a) := a[1]^2 - (x[1]-a[2])^2 - (x[2]-a[3])^2$, where $a[1]$, $a[2]$, and $a[3]$ are parameters meaning the radius, and the center (X,Y) of the cylinder, respectively. All the primitives are combined together using the subtraction operator (\), which is itself defined analytically as discussed in [17] (see Fig. 3 for the associated constructive tree).

Figure 7 (a) shows the results of a typical GP run obtained with a tournament of size 50 (10% of the population) and an initial tree depth of 6 with maximum value 10. It is immediately apparent that GP is able to easily achieve good fitness values, and keep improving them as the evolution proceeds (best value of this run was 20.7). However, the typical GP solution is not parsimonious at all, obtaining good approximations only thanks to an extensive and creative combination of primitives, one that does not reflect the simplicity of the object being modelled. For instance, the run illustrated in the left plot produces a solution containing 46 operations, 47 primitives, and $76 + 218$ parameter elements (arithmetic constants and constants), totalling 387 nodes in the construction tree. The first primitive used in this tree was not the box.

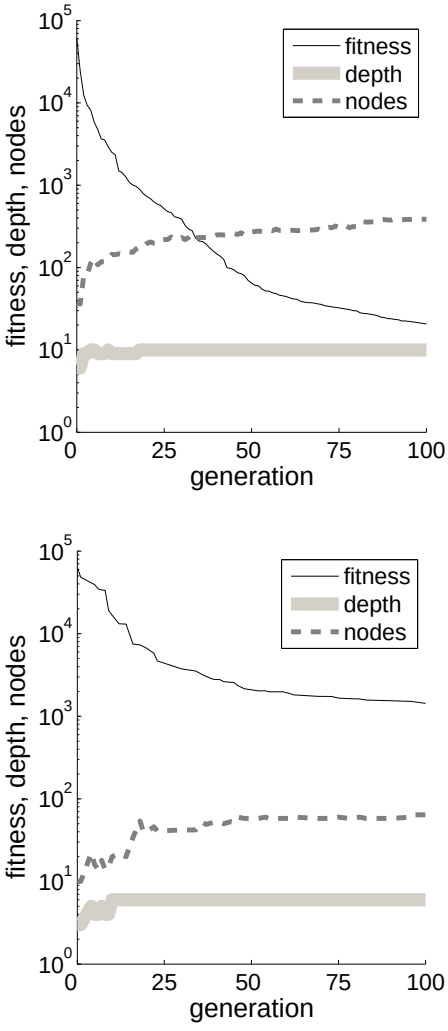
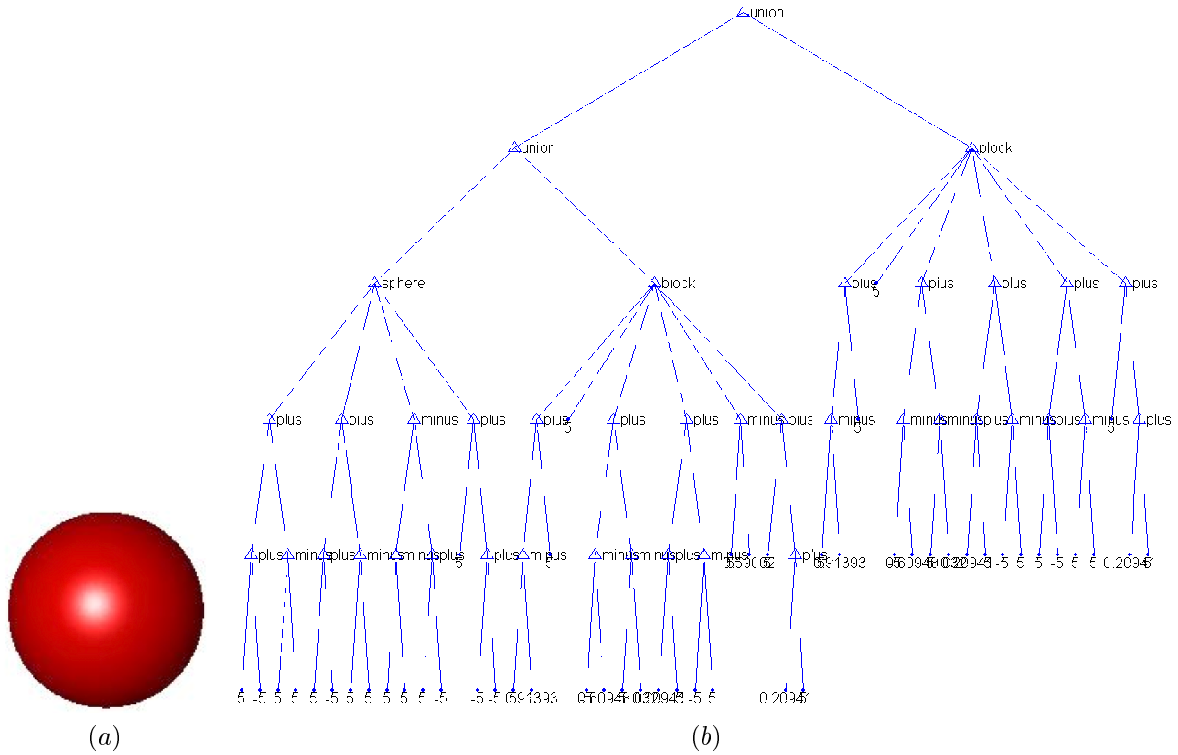


Figure 7. Results of typical GP runs, with high selective pressure and low parsimony pressure *(up)*, and with low selective pressure and high parsimony pressure *(down)*



(a) (b)

Figure 5. Visualisation of the generated FREP model *(a)* **and its associated tree** *(b)*

On the attempt to produce shorter and more realistic solutions, we have increased the parsimony pressure by using an initial tree depth of only 3, with maximum value 6. We have also decreased the selective pressure to allow a larger exploration of the search space, by reducing the tournament size to 2. Figure 7 (b) illustrates a typical run obtained with these parameters. The convergence to better fitness values was much more difficult (best value of this run was 1434.4), and the solution produced was much smaller, but still far from expected, containing 8 operations, 9 primitives, and $10 + 37$ parameter elements (operators and constants), totalling 64 nodes in the construction tree. Even if the fitness function has a low value, the resulting tree is very complex. One post-processing consisting in cutting useless subtrees(such as a sphere with a null radius) in order to simplify the resulting FREP model.

4 Conclusion

We have extended the work presented in [5] for determining and fitting a parameterized FRep model to a 3D point data set. The strongly-typed GP approach is able to produce highly fit solutions in a totally automatic manner. However, the construction trees generated suffer from the extreme lack of parsimony that usually plagues GP evolution, even when parsimony pressure measures are applied.

Further work should be performed in order to make GP operate more like the iterative generation of the construction trees. Starting from very small trees and regular genetic operators to optimize the parameters, other genetic operators could be used to specifically add primitives to the tree.

Real-world objects of higher complexity should be used to test both approaches, reflecting the more realistic conditions where the shape modelling and fitting problem is too difficult to be performed by the user. Under such conditions, the creativeness and flexibility of the GP approach may become essential in producing acceptable results.

Acknowledgement

C. Toinard and C. Rosenberger would like to thank financial support provided by the Conseil Général du Cher.

P.A. Fayolle acknowledges support by Mombusho, the Japanese ministry of research and education.

This work was partially financed by grant SFRH/BD/14167/2003 from Fundação para a Ciência e a Tecnologia, Portugal. We would like to thank the members of the ECOS – Evolutionary and Complex Systems Group (Universidade de Coimbra), in particu-

lar group leader Ernesto Costa, for the valuable ideas and suggestions provided regarding this work.

References

[1] Back, T., Fogel, D.B., Michalewicz, Z. (eds.): Handbook of Evolutionary Computation. IOP Publishing and Oxford University Press, New York and Bristol (1997)

[2] Banzhaf, W., Nordin, P., Keller, R.E., Francone, F.D.: Genetic Programming - An Introduction. Morgan Kaufmann, San Francisco (1998)

[3] Benko, P., Kos, G., Varady, T., Andor, L., Martin, R.: Constrained Fitting in Reverse Engineering. Computer Aided Geometric Design **19** (2002) 173–205

[4] Costantini, F., Toinard, C.: Collaboration and Virtual Early Prototyping Using the Distributed Building Site Metaphor. In: Rahman, S.M.M. (ed.): Multimedia Networking: Technology, Management and Applications (2002) 290–332

[5] Fayolle, P.-A., Rosenberger, C., Toinard, C.: Shape Recovery and Functional Modeling Using Genetic Algorithms. In: Proceedings of IEEE LAVAL VIRTUAL (2004) 227–232

[6] Fayolle, P.-A., Pasko, A., Kartasheva, E., Mirenkov, N.: Shape Recovery Using Functionally Represented Constructive Models. In: Proceedings of SMI 2004 (2004) (in press)

[7] Fisher, R.: Applying Knowledge to Reverse Engineering Problems. In: Proceedings of Geometric Modeling and Processing. IEEE Computer Society (2002) 149–155

[8] Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, Boston, MA (1989)

[9] HyperFun project (2005), http://cis.k.hosei.ac.jp/ F-rep/

[10] Holland, J.H.: Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor (1975)

[11] Houck, C., Joines, J., Kay, M.: A Genetic Algorithm for Function Optimization: A Matlab Implementation. Technical Report NCSU-IE TR 95-09 (1995)

[12] Koza, J.R.: Genetic Programming –On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge, MA (1992)

[13] Luke, S., Panait, L.: Lexicographic Parsimony Pressure. In: Langdon, W.B. *et al.* (eds.): Proceedings of GECCO-2002. Morgan Kaufmann (2002) 829–836

[14] Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag, Berlin (1996)

[15] Michalewicz, Z., Fogel, D.B.: How to Solve It: Modern Heuristics, 2nd edn. Springer, Berlin (2004)

[16] Montana, D.J.: Strongly Typed Genetic Programming. BBN Technical Report #7866 (1994)

[17] Pasko, A., Adzhiev, V., Sourin, A., Savchenko, V.: Function Representation in Geometric Modeling: Concepts, Implementation and Applications. The Visual Computer **11** (1995) 429–446

[18] Robertson, C., Fisher, R., Werghi, N., Ashbrook, A.: An Evolutionary Approach to Fitting Constrained Degenerate Second Order Surfaces. EvoWorkshops (1999) 1–16.

[19] Seo, H., Magnenat-Thalmann, N.: An Example-Based Approach to Human Body Manipulation. Graphical Models **66** (2004) 1–23

[20] Silva, S.: GPLAB – A Genetic Programming Toolbox for MATLAB (2005), http://gplab.sourceforge.net

[21] Silva, S., Costa, E. Dynamic Limits for Bloat Control - Variations on Size and Depth. In: Deb, K., Poli, R., Banzhaf, W. *et al.* (eds.): Proceedings of GECCO-2004. Springer (2004) 666–677

[22] The MathWorks – MATLAB and Simulink for Technical Computing (2005), http://www.mathworks.com/