22es Journées de l'Association Francophone d'Informatique Graphique, Arles, 2009

Un algorithme exact et performant, basé sur les sommets contributeurs, pour le calcul de la somme de Minkowski d'une paire de polyèdres non convexe/convexe

Hichem Barki [1], Florence Denis[1] et Florent Dupont[1]

[1]Université de Lyon, CNRS
Université Lyon 1, LIRIS, UMR5205

Abstract

We present the NCC-CVMS algorithm, an exact and efficient Contributing Vertices-based Minkowski Sum algorithm for the computation of the Minkowski sum of a Non-Convex–Convex pair of closed and two-manifold polyhedra, without decomposition and union steps. First, we generate a reduced superset (with a minimum number of facets) of the Minkowski sum facets through the use of the contributing vertices concept. Secondly, we compute the 2D arrangements induced by the superset triangles intersections. Finally, we obtain the Minkowski sum through the use of two simple properties of the input polyhedra and the Minkowski sum polyhedron itself, i.e. the closeness and the two-manifoldness properties. The NCC-CVMS algorithm is efficient because of the simplifications induced by the use of the contributing vertices concept, the use of 2D arrangements, and the use of the simple properties of closeness and manifoldness to recover the Minkowski sum polyhedron. However, it does not recover eventual polyhedral holes inside the Minkowski sum polyhedron. We implemented our NCC-CVMS algorithm on the base of CGAL and used exact number types.

Nous présentons l'algorithme NCC-CVMS : un algorithme exact et efficace, basé sur la notion de sommets contributeurs, pour le calcul de la Somme de Minkowski (SM) d'une paire de polyèdres non convexe/convexe fermés et des 2-variétés, sans décomposition en convexes et sans calcul d'union. Premièrement, nous générons un sur-ensemble réduit (avec un nombre minimum de facettes) des facettes du polyèdre SM en exploitant la notion de sommets contributeurs. Deuxièmement, nous calculons les arrangements 2D induits par les intersections des triangles de ce sur-ensemble. Enfin, nous obtenons la SM par l'utilisation de deux propriétés simples des polyèdres d'entrée et du polyèdre SM lui-même, à savoir la propriété de fermeture et celle de 2-variété. L'algorithme NCC-CVMS est performant grâce aux simplifications induites par l'utilisation de la notion de sommets contributeurs, l'utilisation des arrangements et l'utilisation des propriétés simples de fermeture et de 2-variété afin de reconstruire le polyèdre SM. Toutefois, notre algorithme ne reconstruit pas les trous éventuellement présents à l'intérieur du polyèdre SM. Nous avons implanté notre algorithme NCC-CVMS en utilisant la bibliothèque CGAL et les types de nombres exacts.

1. Introduction

La SM de deux ensembles $\mathcal{A}$ et $\mathcal{B}$ dans $\mathbb{R}^n$ est définie comme l'addition vectorielle de tous les éléments a et b provenant de $\mathcal{A}$ et de $\mathcal{B}$ respectivement : $\mathcal{A} \oplus \mathcal{B} = \{a+b | a \in \mathcal{A}, b \in \mathcal{B}\}$. Une autre définition stipule que la SM de deux ensembles $\mathcal{A}$ et $\mathcal{B}$ est obtenue en balayant tous les points de $\mathcal{A}$ par $\mathcal{B}$, c'est à dire en translatant $\mathcal{B}$ de sorte que son origine (le point initial commun de tous ses vecteurs de position) passe par tous les points de $\mathcal{A}$ et en calculant l'union de tous les points résultants : $\mathcal{A} \oplus \mathcal{B} = \bigcup_{a \in \mathcal{A}} \mathcal{B}_a$, où $\cup$ dénote l'opérateur d'union et $\mathcal{B}_a$ est l'ensemble $\mathcal{B}$ translaté par un vecteur de position a.

Notre objectif est de calculer le polyèdre SM $S = A \oplus B$. Les polyèdres A et B sont les représentations surfaciques respectives des ensembles $\mathcal{A}$ et $\mathcal{B}$ dans $\mathbb{R}^3$ ($A = \partial\mathcal{A}$ et $B = \partial\mathcal{B}$). Il est clair que $\mathcal{A}$ et $\mathcal{B}$ sont complètement définis par leurs

surfaces A et B. De plus, la SM $\mathcal{A} \oplus \mathcal{B}$ est aussi complètement définie par le polyèdre $S = A \oplus B = \partial(\mathcal{A} \oplus \mathcal{B})$ (sa représentation surfacique correspondante). Donc, l'équation basé sur le balayage devient :

$$S = A \oplus B = \bigcup_{a \in A} B_a \tag{1}$$

Ce qui signifie que pour calculer le polyèdre SM S, il suffit seulement de balayer la surface du polyèdre A par B et de prendre la surface de l'union de tous les points résultants comme polyèdre SM (voir Fig. 1).

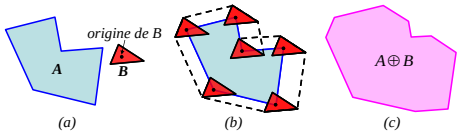

Figure 1: *(a) Deux polygones A et B. (b) Balayage de la surface de A par B. (c) Le polygone SM $A \oplus B$.*

Un polyèdre est dit fermé et de 2-variété si chacune de ses arêtes est incidente à exactement deux de ces facettes. Si A et B sont fermés et des 2-variétés, le polyèdre SM $S = A \oplus B$ sera aussi fermé et de 2-variété parce qu'il correspond à l'union des translations d'un polyèdre fermé et de 2-variété B et parce que ces translations sont définies par la surface extérieure d'un polyèdre fermé et de 2-variété A.

Nous avons introduit la notion de sommets contributeurs, qui est liée à la définition par balayage de la SM, pour la première fois dans [BDD09a]. Cette notion nous a permis de développer l'algorithme CVMS (Contributing Vertices-based Minkowski Sum) : un algorithme exact, efficace et gérant les cas dégénérés pour le calcul de la SM de polyèdres convexes. Les expérimentations effectuées ont montré que notre algorithme CVMS est plus performant que les algorithmes de l'état de l'art présentés dans [FH07, HKM07, Wei]. Dans [BDD09b], nous avons repris la notion de sommets contributeurs et avons proposé l'algorithme CVMS pour le calcul de la SM d'une paire de polyèdres non-convexe/convexe sans repli. Premièrement, nous avons généré un sur-ensemble des facettes la SM. Deuxièmement, nous avons extrait la surface exacte du polyèdre SM en calculant des enveloppes 3D du sur-ensemble réduit. Cependant, cette approche est limitée aux polyèdres sans repli—polyèdres dont la surface extérieure est entièrement recouvrable à partir de trois projections orthographiques définies par rapport à trois vecteurs de base orthogonaux dans $\mathbb{R}^3$.

L'algorithme que nous proposons dans ce papier réutilise le sur-ensemble déjà généré pour une paire de polyèdres non convexe/convexe. Il ne se limite pas aux polyèdres sans repli et il est plus efficace que l'approche proposée dans [BDD09b]. Il est nommé l'algorithme NCC-CVMS. Bien que l'abréviation CVMS signifie "Contributing Vertices-based Minkowski Sum" (comme pour les deux autres algorithmes), le préfixe NCC qui signifie "Non-Convex–Convex" le distingue dans ce travail.

L'algorithme NCC-CVMS que nous proposons dans ce papier réutilise le sur-ensemble déjà généré. Il procède comme suit : (1) il calcule le sur-ensemble réduit des facettes de la SM grâce à l'utilisation de la notion de sommets contributeurs, (2), il calcule les arrangements planaires ou arrangements 2D induits par les intersections entre les triangles du sur-ensemble et (3) il utilise les propriétés de fermeture et de 2-variété afin d'extraire les facettes du polyèdre SM et d'ignorer les autres (facettes situées à l'intérieur du polyèdre SM). Dans l'étape (2), nous utilisons un arrangement 2D pour chaque triangle du sur-ensemble, ce qui nous évite la gestion difficile des arrangements 3D et augmente les performances de l'algorithme. Dans l'étape (3), nous partons d'une facette germe dont l'appartenance au polyèdre SM est garantie. Ensuite nous examinons ses facettes voisines, ajoutons celles qui respectent la propriété de 2-variété au polyèdre SM et éliminons les autres facettes qui ne satisfont pas cette propriété. Ce processus est répété en considérant les facettes de la SM récemment trouvées comme facettes germes jusqu'à ce que la propriété de fermeture soit satisfaite, c'est à dire jusqu'à ce que le polyèdre résultant soit fermé. En procédant ainsi, nous évitons le calcul difficile des nombres de tours des subdivisions des arrangements (qui constitue un inconvénient majeur de l'approche par convolution [BGRR96]) et réduisons aussi la complexité de l'algorithme.

En section 2, nous présenterons l'état de l'art de la SM de polyèdres. Après cela, nous décrirons dans la section 3 la partie de l'algorithme NCC-CVMS qui permet l'extraction de la SM à partir du sur-ensemble de facettes. La partie de génération du sur-ensemble de facettes de la SM n'est pas discutée, le lecteur pourra se référer à [BDD09b] pour plus de détails la concernant. Enfin, nous présenterons certains détails d'implantation, effectuerons des expérimentations et présenterons des résultats obtenus.

2. État de l'art

La SM de deux polyèdres convexes A et B s'obtient en effectuant l'addition vectorielle de tous les points de A et B et en calculant l'enveloppe convexe du nuage de points résultants. Ce processus a une complexité en temps de $O(mn)$ pour deux polyèdres ayant m et n facettes. Pour deux polyèdres non convexes, l'approche la plus commune est basée sur une décomposition en convexes. Elle a une complexité en temps de $O(m^3n^3)$. Le calcul de la SM se fait en décomposant chaque polyèdre non convexe en polyèdres convexes, en calculant les SM partielles de toutes les paires de polyèdres convexes et en effectuant l'union de ces SM partielles. L'inconvénient majeur de cette approche est l'étape d'union qui est gourmande en termes de temps de calcul.

Récemment, Hachenberger [Hac07] a présenté la première implantation exacte et robuste, basée sur les polyèdres Nef, de l'approche par décomposition en convexes pour le calcul de la SM de polyèdres ayant m et n facettes en un temps $O(m^3n^3)$. Il a implanté un algorithme optimal de décomposition en convexes similaire à celui proposé par Chazelle [Cha81]. Il décompose les polyèdres convexes en insérant des plans afin d'éliminer les arêtes de repli jusqu'à ce que toutes les arêtes de repli soient traitées. Il calcule ensuite l'union des SM partielles en utilisant les polyèdres Nef qui gèrent correctement les opérations booléennes.

Pour éviter les inconvénients liés à l'approche par décomposition, Varadhan et Manocha [VM06] ont approximé l'union des SM partielles à l'aide d'une grille de voxels subdivisée adaptativement tout en garantissant la même topologie de la SM. Récemment, Lien [Lie07] a utilisé une représentation par points au lieu de celle basée sur des maillages pour le calcul de la SM. Il a utilisé trois filtres de détection de collision, de normales et d'octree afin de recouvrer les points se trouvant sur la surface du polyèdre SM.

Ghosh [Gho93] a présenté un cadre unifié pour le calcul de la SM. Il représente les polyèdres dans un espace dual—le diagramme de pente, fusionne les diagrammes de pente de deux polyèdres, et calcule le polyèdre SM à partir des intersections des deux diagrammes. Cependant, les implantations existantes pour les algorithmes basés sur les diagrammes de pente sont restreintes aux polyèdres convexes. D'autres variantes des approches par diagrammes de pente peuvent être trouvées dans [BDD09a, FH07].

Guibas et al. [GS87] ont défini l'opération de convolution de tracés planaires 2D. Basch et al. [BGRR96] l'ont étendu aux tracés polyédriques et l'ont utilisé pour générer un sur-ensemble de la SM. Ils ont utilisé des arrangements 3D afin d'extraire le polyèdre SM du sur-ensemble.

En proposant l'algorithme NCC-CVMS, nous apportons une solution exacte et efficace au problème du calcul de la SM de polyèdres tout en évitant les défauts des autres approches, à savoir : (1) la décomposition en convexes et le calcul difficile de l'union de polyèdres et (2) la gestion compliquée des arrangements 3D et des nombres de tours utilisées en convolution.

3. Extraction du polyèdre SM à partir du sur-ensemble

Dans ce qui suit, nous considérons un polyèdre non convexe, fermé et de 2-variété A ainsi qu'un autre polyèdre convexe, fermé et de 2-variété B. Nous dénotons par $F = \{f_1, f_2, \ldots, f_n\}$ le sur-ensemble de facettes de la SM $S = A \oplus B$.

Les intersections entre les facettes du sur-ensemble sont gérées à l'aide d'arrangements 2D. Pour cela, nous commencerons tout d'abord par la description du mécanisme permettant le passage entre caractéristiques 3D (facettes 3D, segments 3D et points 3D) et caractéristiques équivalentes en 2D (facettes d'arrangements 2D, segments d'arrangements 2D et sommets ou points d'arrangements 2D).

3.1. Correspondance entre le sur-ensemble de facettes de la SM et les arrangements 2D

Considérons une facette arbitraire f_i du sur-ensemble F ayant une normale $n_i(x_{n_i}, y_{n_i}, z_{n_i})$ et un plan d'appui P_i. Pour représenter la facette 3D f_i par une facette d'un arrangement 2D $arr(f_i)$, nous avons besoin d'une projection bijective qui associe un polygone 2D (une facette d'arrangement 2D) à chaque polygone 3D (une facette du sur-ensemble). Une solution efficace en termes de temps de calcul et n'engendrant pas de cas dégénérés (telle que la projection d'un polygone 3D en un segment 2D) consiste à comparer les valeurs absolues des coordonnées x_{n_i}, y_{n_i} et z_{n_i} de la normale à f_i, d'ignorer celle ayant la valeur absolue la plus grande et de garder les valeurs des deux autres coordonnées. Étant donné un point 3D $p_{f_i}(x,y,z)$ appartenant à f_i, le point de l'arrangement 2D lui correspondant $p_{arr(f_i)}(x_{arr}, y_{arr})$ est alors calculé comme suit :

$$x_{arr} = y \text{ and } y_{arr} = z, \ \ |x_{n_i}| = max(|x_{n_i}|, |y_{n_i}|, |z_{n_i}|) \quad (2)$$

$$x_{arr} = z \text{ and } y_{arr} = x, \ \ |y_{n_i}| = max(|x_{n_i}|, |y_{n_i}|, |z_{n_i}|) \quad (3)$$

$$x_{arr} = x \text{ and } y_{arr} = y, \ \ |z_{n_i}| = max(|x_{n_i}|, |y_{n_i}|, |z_{n_i}|) \quad (4)$$

où x_{arr} et y_{arr} représentent les coordonnées respectives x et y de $p_{arr(f_i)}$ dans le système de coordonnées de l'arrangement 2D $arr(f_i)$.

La projection bijective inverse du point 2D vers le point 3D correspondant peut être effectuée en substituant les valeurs des coordonnées du point 2D dans l'équation $a_ix + b_iy + c_iz + d_i = 0$ du plan d'appui P_i de la facette f_i afin d'obtenir la valeur de la troisième coordonnée.

3.2. L'algorithme d'extraction de la surface du polyèdre SM en détail

L'algorithme NCC-CVMS commence par la génération du sur-ensemble réduit des facettes de la SM [BDD09b] dont l'enveloppe 3D est le polyèdre SM lui-même. Cependant, certaines parties des facettes du sur-ensemble se trouvent à l'intérieur du polyèdre SM. Par conséquent, un filtrage du sur-ensemble est nécessaire afin d'enlever ces parties, ce qui implique la gestion des intersections entre les facettes du sur-ensemble.

Nous commencerons par donner un plan de notre algorithme NCC-CVMS. Ensuite, nous expliquerons chacune de ces étapes. Donc, étant donné une paire de polyèdres non convexe/convexe A et B, nous désignons par F le sur-ensemble des facettes de la SM, par *MS_stack* la pile utilisée pour manipuler les facettes appartenant à la SM (contenant des éléments de type "facette d'arrangement

2D") et par top_{MS_stack} une variable mémorisant la tête de la pile. L'algorithme NCC-CVMS procède comme suit :

Algorithme 1 : L'agorithme NCC-CVMS pour une paire de polyèdres non convexe/convexe

Entrées : Une paire de polyèdres non convexe/convexe A et B
Sorties : Le polyèdre SM $S = A \oplus B$

1 Générer le sur-ensemble réduit F des facettes de la SM;
2 Fusionner les facettes "coplanaires conjointes par l'intérieur" du sur-ensemble F;
3 Trianguler le sur-ensemble F;
4 Calculer les boites Iso-orientées englobant chaque triangle du sur-ensemble F et calculer les intersections de ces boites;
5 Calculer les intersections entre les triangles du sur-ensemble F dont les boites englobantes s'intersectent;
6 Construire un arrangement 2D pour chaque triangle du sur-ensemble F à partir des segments d'intersections (subdivisions des triangles du sur-ensemble en sous-facettes);
7 Calculer la facette germe du sur-ensemble F;
8 Empiler la sous-facette germe dans *MS_stack*;
9 **répéter**
10 $top_{MS\_stack} \leftarrow MS\_stack.top$;
11 *MS_stack.pop* (dépiler *MS_stack*);
12 Trouver les voisins de top_{MS_stack} (facettes d'arrangements 2D associées aux sous-facettes du sur-ensemble qui sont incidentes aux arêtes de la sous-facette associée à la facette d'arrangement 2D top_{MS_stack});
13 Parmi les sous-facettes voisines, déterminer celles appartenant à la surface du polyèdre SM en utilisant les propriétés de fermeture et de 2-variété;
14 Étiqueter les sous-facettes de la SM et les sous-facettes n'appartenant pas à la SM correctement;
15 Mettre à jour *MS_stack* (empiler les facettes voisines étiquetées comme appartenant à la SM dans *MS_stack*);
16 **jusqu'à** *(MS_stack est vide)* ;
17 Construire le polyèdre SM à partir des facettes des arrangements 2D;

Étape (1) Générer le sur-ensemble réduit F des facettes de la SM de A et B : Plus de détails peuvent être consultés dans [BDD09b].

Étape (2) Fusionner les facettes "coplanaires conjointes par l'intérieur" du sur-ensemble F : Deux facettes coplanaires (du même plan d'appui) sont dites "conjointes par l'intérieur" s'il existe un point commun appartenant à l'intérieur des deux facettes en même temps. L'intérieur d'une facette est défini par l'ensemble de points de cette facette excepté ceux des arêtes et des sommets incidents à elle. Le principe de cette étape est de fusionner (ou calculer l'union) de toutes les facettes "coplanaires conjointes par l'intérieur" en une seule facette (voir Fig. 2). Cette étape ne change pas la propriété de fermeture du sur-ensemble F et permet de simplifier l'étape (5) (calcul des intersections entre triangles) comme nous le verrons par la suite.

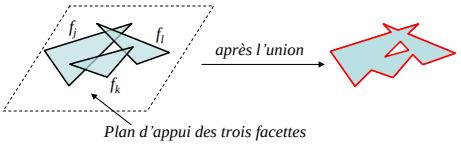

Figure 2: *Fusion des facettes coplanaires conjointes par intérieur f_j, $f_k$et f_l en une seule facette.*

Étape (3) Trianguler le sur-ensemble F Afin de filtrer le sur-ensemble F, nous devons gérer les intersections entre les facettes polygonales f_1, f_2, ..., f_n après l'étape de fusion. Cependant, ces facettes polygonales 3D peuvent être convexes ou non convexes. De plus, certaines facettes peuvent avoir des trous dûs à l'étape de fusion (voir Fig. 2). Tous ces facteurs rendent le calcul des intersections entre ces facettes difficile, coûteux et sujet à des dégénérescences. Afin de faciliter le calcul des intersections, nous triangulons toutes les facettes afin de réduire le problème à celui du calcul d'intersections entre triangles 3D. Après cette étape, le sur-ensemble F sera composé de n_T triangles : $F = \{t_1, t_2, \ldots, t_{n_T}\}$, $(n_T \geq n)$ où n est le nombre de facettes polygonales du sur-ensemble F et n_T est le nombre de triangles de F restants à la fin de cette étape.

Étape (4) Calculer les boites Iso-orientées englobant chaque triangle du sur-ensemble F et calculer les intersections de ces boites : Cette étape vise à accélérer le calcul des intersections entre les triangles du sur-ensemble F, car de tels calculs sont coûteux en pratique. L'intersection exacte entre deux triangles n'est calculée que si les boites englobantes des deux triangles s'intersectent. Pour cela, nous utilisons l'algorithme simple et efficace proposé dans [ZE02] qui opère sur des boites englobantes Iso-orientées (boites alignées aux axes cartésiens).

Étape (5) Calculer les intersections entre les triangles du sur-ensemble F dont les boites englobantes s'intersectent : Pour ce faire, nous utilisons une version légèrement modifiée de "l'approche par chevauchement d'intervalles" proposée dans [MÖ97].

La fusion des facettes "coplanaires conjointes par l'intérieur" (étape (2)) est importante afin d'éviter d'avoir à calculer des intersections de triangles coplanaires (cas dégénérés). Ce type de calcul est coûteux et donne un polygone 3D au lieu d'un segment utilisable pour la construction

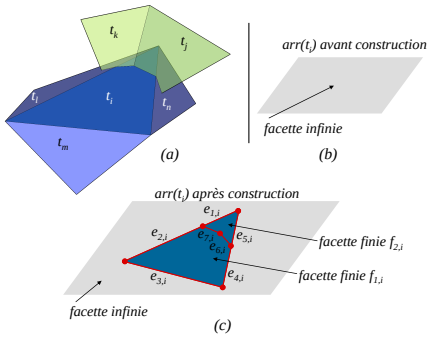

Figure 3: *(a) Une liste de triangles du sur-ensemble F intersectant le triangle t_i. (b) L'arrangement planaire $arr(t_i)$ avant l'insertion des segments. Il se compose d'une facette infinie. (c) L'arrangement $arr(t_i)$ après l'insertion des segments $e_{1,i}$, $e_{2,i}$, ..., $e_{7,i}$.*

d'arrangements 2D.

Étape (6) Construire un arrangement 2D pour chaque triangle du sur-ensemble F à partir des segments d'intersections : Cette étape consiste à construire des arrangements 2D pour les n_T triangles du sur-ensemble F. Pour un triangle t_i, $i = 1, 2, \ldots, n_T$ de F, l'arrangement 2D correspondant est la subdivision du plan par l'insertion des segments représentants les arêtes du triangle t_i d'une part et par ceux qui résultant de l'intersection de t_i avec les autres triangles du sur-ensemble F (voir Fig. 3 pour un exemple).

Après cette étape, nous avons un arrangement planaire arr (t_i) associée à chaque triangle t_i, $1 \leq i \leq n_T$ du sur-ensemble F. Ces arrangements planaires définissent des subdivisions des triangles du sur-ensemble en sous-facettes, chaque sous-facette soit appartient à la surface du polyèdre SM, soit elle se trouve à l'intérieur de ce polyèdre.

Étape (7) Calculer la facette germe en trouvant le point lexicographiquement plus petit ou plus grand du sur-ensemble : L'idée est de partir d'une sous-facette germe dont l'appartenance au polyèdre SM est garantie et de parcourir son voisinage afin de trouver les autres facettes de la SM. Puisque le sur-ensemble F est l'enveloppe du polyèdre *SM* qui est à son tour un polyèdre fermé, toutes les sous-facettes à l'intérieur du sur-ensemble F sont nécessairement invisibles à partir de l'extérieur de celui-ci, ce qui implique que toutes les sous-facettes visibles de l'extérieur du sur-ensemble F sont des facettes du polyèdre SM. Donc, il suffit de calculer le point lexicographiquement plus petit ou plus grand (qui est nécessairement visible de l'extérieur du sur-ensemble) et de prendre parmi les sous-facettes incidentes à ce point, celle qui laisse toutes les autres voisines de ce point sur le côté négatif de son plan

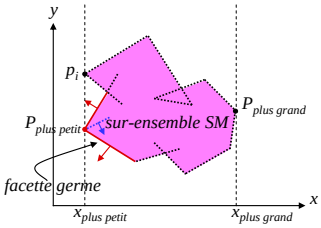

Figure 4: *Le choix de la face (arête) germe d'un sur-ensemble de la SM en 2D. La face germe peut être prise comme l'une ou l'autre des deux dessinées par des lignes continues.*

d'appui comme facette germe (voir Fig. 4).

Étape (8) Empiler la sous-facette germe de l'arrangement 2D dans *MS_stack* : Cette étape permet d'initialiser *MS_stack* avec la sous-facette germe.

Étapes (9) à (16) Rechercher les voisins de la sous-facette germe et déterminer les facettes de la surface du polyèdre SM :

Deux types d'étiquettes sont associées à chaque sous-facette du sur-ensemble F. Le premier type reflète l'appartenance d'une sous-facette à la surface du polyèdre SM (valeurs "sous-facette SM" ou 'sous-facette non-SM"). Le deuxième type d'étiquettes indique si une sous-facette a été déjà empilée ou pas encore dans *MS_stack* (valeurs "déjà empilée" et "pas encore empilée"). Il permet d'éviter de parcourir une sous-facette plus d'une fois. Avant l'utilisation de la pile *MS_stack* (avant l'exécution de la ligne 8 de l'algorithme 1), toutes les sous-facettes sont étiquetées comme "sous-facette non-MS" et comme "pas encore empilée". Durant les étapes (9) à (16), chaque sous-facette empilée sera systématiquement étiquetée comme "sous-facette MS" et comme "déjà empilée".

Considérons la facette d'arrangement top_{MS_stack} d'une itération arbitraire de la boucle "répéter-jusqu'à" de l'algorithme 1 et supposons que cette facette soit associée à une sous-facette 3D facette $f_{top_{MS\_stack}}$ appartenant à la subdivision du triangle t_i. Il est clair que $f_{top_{MS\_stack}}$ appartient à la surface du polyèdre SM puisqu'elle a déjà été empilée. De plus, considérons un segment 3D e_j de la sous-facette $f_{top_{MS\_stack}}$ résultant de l'intersection de plusieurs triangles avec t_i (voir Fig. 5.a). Parmi les sous-facettes des triangles qui intersectent t_i à l'arête e_j, seule une d'entre elles vérifie la propriété de 2-variété. La sous-facette qui doit être choisie comme sous-facette de la SM est la sous-facette du plan d'appui qui forme le plus petit angle par rapport au plan d'appui de t_i parmi toutes les autres sous-facettes des autres triangles (voir Fig. 5.b). Ce critère d'angles de plans d'appui garantit que les autres sous-facettes non-choisies se situent

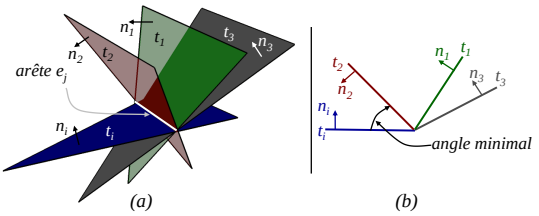

Figure 5: *Le critère des angles des plans d'appui. (a) Les triangles t_1, t_2 et t_3 sont incidents à la même arête e_j de t_i. (b) L'angle minimal est celui entre le plan d'appui de t_i et celui de t_2. La sous-facette SM incidente à e_j qui doit être choisie est une sous-facette du triangle t_2 située sur le côté positif du plan d'appui de t_i.*

à l'intérieur du polyèdre SM. Pour l'exemple représenté en Fig.5, la sous-facette qui doit être choisie est une des deux sous-facettes du triangle t_2. Cependant, deux sous-facettes du triangle t_2 sont incidentes à l'arête e_j et forment le même angle minimal par rapport au plan d'appui de t_i. La sous-facette à choisir comme sous-facette de la SM sera celle de t_2 située sur le côté positif du plan d'appui de t_i. Donc, pour cette itération arbitraire de la boucle "répéter-jusqu'à", la sous-facette choisie est étiquetée comme "sous-facette SM", empilée dans *MS_stack* et étiquetée comme "déjà empilée". Toutes les autres sous-facettes incidentes à l'arête e_j sont étiquetées comme "sous-facette non-SM". La boucle est répétée jusqu'à ce que la pile *MS_stack* soit vide, c'est à dire jusqu'à ce que la propriété de fermeture du polyèdre résultat soit satisfaite.

Étape (17) Construire le polyèdre SM à partir des facettes des arrangements 2D : Pour ce faire, ils suffit tout simplement de chercher toutes les facettes des arrangement 2D qui sont étiquetées comme "sous-facette SM" et le les projeter sur les plans d'appui des triangles correspondants du sur-ensemble F afin d'obtenir le polyèdre SM.

4. Implantation et résultats expérimentaux

Dans cette section, nous donnerons certains détails de notre implantation et effectuerons des expérimentations ainsi que des comparaisons entre plusieurs algorithmes de calcul de la SM.

4.1. Implantation

L'algorithme NCC-CVMS a été implanté en C++ et avec la bibliothèque CGAL (Computational Geometry Algorithms Library) [CGA]. Nous avons utilisé les types de nombres exacts fournis par la librairie GNU Multi Precision [GMP] sous CGAL et les avons accélérés en utilisant le "lazy kernel adapter" [FP06]. Notons aussi l'utilisation de l'implantation des polyèdres Nef [Hac07] fournie dans CGAL pour la comparaison des temps d'exécution.

4.2. Résultats expérimentaux et comparaisons

Pour nos tests, nous avons comparé l'algorithme NCC-CVMS à l'approche par décomposition en convexes basée sur les polyèdres Nef proposée dans [Hac07]. Cette approche est à notre connaissance la seule à pouvoir calculer la SM de polyèdres non convexes d'une manière exacte. Tous les tests ont été effectuées sur un PC équipé d'un processeur Intel Core 2 Duo 2,2 GHz et d'une mémoire vive de 4 Go.

Nous avons calculé les polyèdres SM de plusieurs paires de polyèdres non convexe/convexe en utilisant notre algorithme NCC-CVMS ainsi que l'algorithme par décomposition basé sur les polyèdres Nef de Hachenbegrer [Hac07] (implanté dans CGAL). Les temps d'exécution sont indiqués dans le tableau 1. Les tailles des polyèdres A et B (nombre de facettes) ainsi que les tailles des décompositions en convexes (nombre de polyèdres convexes) sont également données dans le tableau 1.

Table 1: *Comparison des temps d'exécution de l'algorithme NCC-CVMS et de l'approche par décomposition basée sur les polyèdres Nef.*

Opérandes		Polyèdres Nef		NCC-CVMS
A	B	Dec.conv.A	Nef (s)	NCC-CVMS (s)
Grate1(540)	Tr.Tetra.(8)	47	**20.547**	**15.171**
Grate1(540)	Sphere(500)	47	**272.406**	**107.968**
Bunny(1500)	Cube(6)	869	**733.266**	**21.720**
Grate2(942)	Rh.Dodeca.(12)	230	**114.125**	**21.031**
Grate2(942)	Sphere(500)	230	**2052.890**	**280.063**
Dinausor(5000)	Cube(6)	3193	**2842.48**	**107.546**

Dec.comp.A - Le nombre de polydères convexes résultant de la décomposition en convexes de A.
Tr.Tetra. - TruncatedTetrahedron, Rh.Dodeca. - RhombicDodecahedron.

À partir des temps d'exécution reportés dans le tableau 1, il est clair que notre algorithme NCC-CVMS est plus rapide que celui basé sur les polyèdres Nef. Cette différence de performance est d'autant plus grande pour des modèles engendrant un grand nombre de polyèdres convexes après décomposition (tels que les modèles *Dinausor* et *Bunny*). Cependant, l'approche par décomposition basée sur les polyèdres Nef gère des paires de polyèdres non convexes qui ne sont pas gérés par notre algorithme NCC-CVMS.

Pour une paire de polyèdres non convexe/convexe sans repli A et B où A est le modèle *Star* ayant 24 facettes et B est le modèle *Sphere* ayant 500 facettes, l'algorithme NCC-CVMS calcule le polyèdre SM en 9.673 secondes tandis que celui que nous avons proposé dans [BDD09b] le calcule en 42.235 secondes, ce qui montre le gain de performance obtenu à l'aide de l'algorithme NCC-CVMS par rapport à notre ancien algorithme. De plus, l'algorithme NCC-CVMS n'est pas restreint aux polyèdres sans repli.

L'algorithme NCC-CVMS gère correctement les polyèdres de taille élevée (ayant des milliers ou des dizaines de milliers de facettes, voir tableau 1). De plus, il gère aussi le changement de genre d'une manière robuste. À titre d'exemple, le modèle *Wrench* (polyèdre A) montré en Fig.

6.d a un genre 4, tandis que le polyèdre SM $Wrench \oplus RhombitruncatedIcosahedron$ a un genre 0 (voir aussi Fig. 6.d). Les modèles avec des genres supérieurs sont également bien traités. Le modèle *Knot* et la SM $Knot \oplus Sphere$ montrés en Fig. 6.a ont un genre 9.

Certains polyèdres SM calculés par l'algorithme NCC-CVMS sont donnés en Fig. 6.

5. Conclusion et perspectives

Nous avons présenté l'algorithme NCC-CVMS, un nouvel algorithme basé sur la notion de sommets contributeurs, pour le calcul de la SM d'une paire de polyèdres non convexe/convexe. Notre algorithme commence par la génération d'un sur-ensemble réduit des facettes de la SM. Ensuite, il calcule les intersections entre les triangles du sur-ensemble et les gère à l'aide d'arrangements 2D. Finalement, en démarrant avec une facette germe, il parcourt son voisinage et extrait les facettes de la surface du polyèdre SM en utilisant les propriétés de fermeture et de 2-variété.

Nous avons implanté notre algorithme en utilisant des types de nombres exacts. Les expérimentations effectuées ont montré que l'algorithme NCC-CVMS est plus performant que la méthode par décomposition basée sur les polyèdres Nef présentée dans [Hac07] ainsi que l'algorithme proposé dans [BDD09b].

Nous travaillons sur la modification de notre algorithme afin de détecter les trous polyédriques éventuellement présents à l'intérieur du polyèdre SM et sur sa généralisation pour une paire de polyèdres non convexes.

References

[BDD09a] BARKI H., DENIS F., DUPONT F. : Contributing vertices-based minkowski sum computation of convex polyhedra. *Comput. Aided Des. 41*, 7 (2009), 525 – 538.

[BDD09b] BARKI H., DENIS F., DUPONT F. : Contributing vertices-based minkowski sum of a non-convex polyhedron without fold and a convex polyhedron. In *IEEE International Conference on Shape Modeling and Applications (SMI'09).* (Beijing, China, June 2009), IEEE Computer Society.

[BGRR96] BASCH J., GUIBAS L., RAMKUMAR G., RAMSHAW L. : Polyhedral tracings and their convolution. In *Proc. of 2nd Workshop on the Algorithmic Foundations of Robotics* (1996), pp. 171–184.

[CGA] CGAL, Computational Geometry Algorithms Library. http ://www.cgal.org.

[Cha81] CHAZELLE B. : Convex decompositions of polyhedra. In *STOC '81 : Proc. of the Thirteenth Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 1981), ACM, pp. 70–79.

[FH07] FOGEL E., HALPERIN D. : Exact and efficient construction of Minkowski sums of convex polyhedra with applications. *Comput. Aided Des. 39*, 11 (2007), 929–940.

[FP06] FABRI A., PION S. : A generic lazy evaluation scheme for exact geometric computations. In *Proc. 2nd Library-Centric Software Design* (2006).

[Gho93] GHOSH P. : A unified computational framework for Minkowski operations. *Comput. Graph. 17*, 4 (1993), 357–378.

[GMP] GNU MP, the GNU MP Bignum Library. http ://gmplib.org.

[GS87] GUIBAS L., SEIDEL R. : Computing convolutions by reciprocal search. *Discrete Comput. Geom. 2* (1987), 175–193.

[Hac07] HACHENBERGER P. : Exact Minkowski sums of polyhedra and exact and efficient decomposition of polyhedra in convex pieces. In *Proc. 15th Annual European Symposium on Algorithms* (2007), pp. 669–680.

[HKM07] HACHENBERGER P., KETTNER L., MEHLHORN K. : Boolean operations on 3d selective nef complexes : data structure, algorithms, optimized implementation and experiments. *Comput. Geom. Theory Appl. 38*, 1-2 (2007), 64–99.

[Lie07] LIEN J.-M. : Point-based Minkowski sum boundary. In *PG '07 : Proc. of the 15th Pacific Conference on Computer Graphics and Applications (PG'07)* (Washington, DC, USA, 2007), IEEE Computer Society, pp. 261–270.

[MÖ97] MÖLLER T. : A fast triangle-triangle intersection test. *journal of graphics tools 2*, 2 (1997), 25–30.

[VM06] VARADHAN G., MANOCHA D. : Accurate Minkowski sum approximation of polyhedral models. *Graph. Models 68*, 4 (2006), 343–355.

[Wei] WEIBEL C. : Minkowski sums. http ://roso.epfl.ch/cw/poly/public.php.

[ZE02] ZOMORODIAN A., EDELSBRUNNER H. : Fast software for box intersections. *International Journal of Computational Geometry and Applications 12*, 1-2 (2002), 143–172.

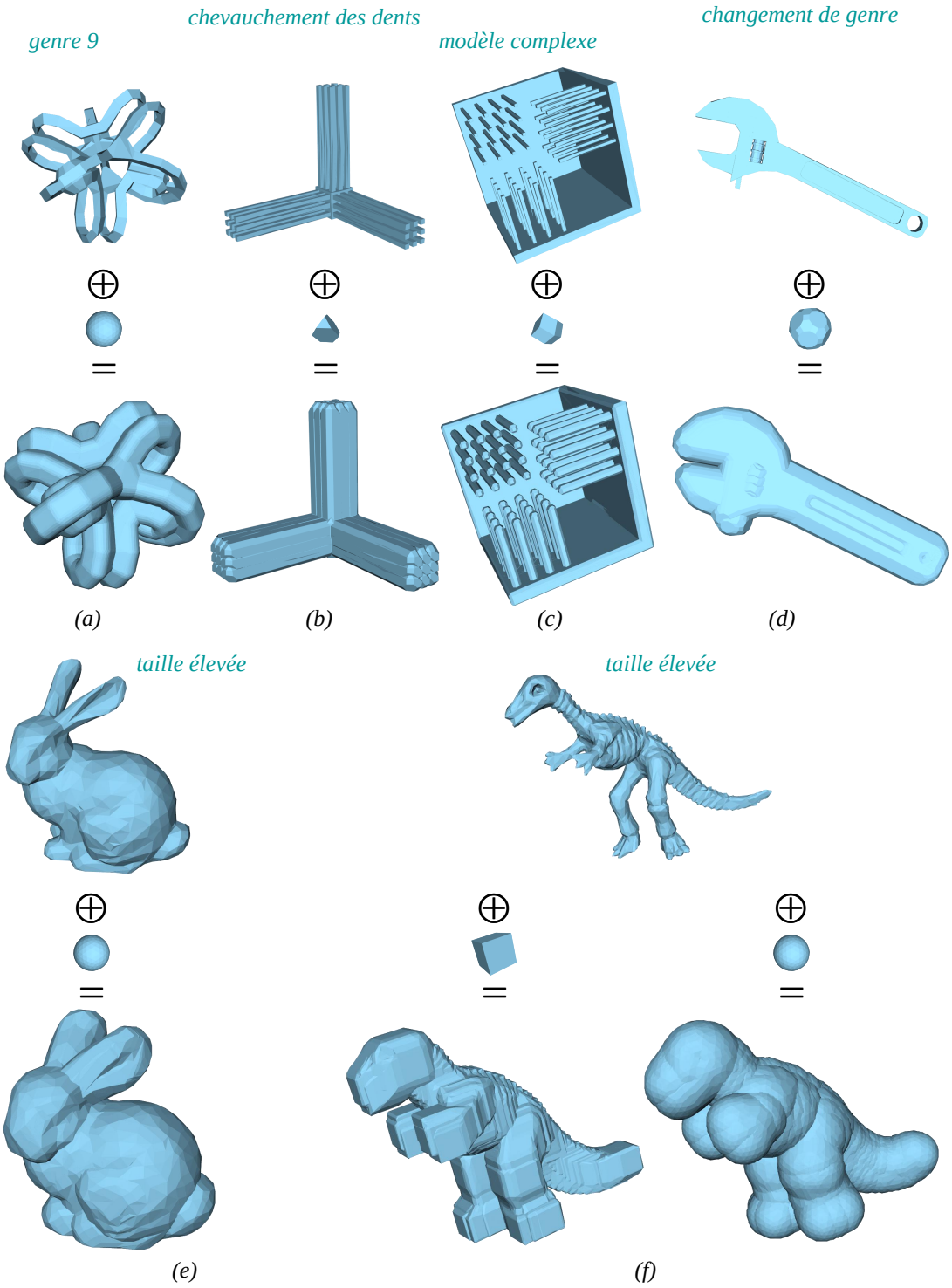

Figure 6: *Polyèdres SM calculés par l'algorithme NCC-CVMS. De haut en bas de chaque sous-figure : le polyèdre A, le polyèdre B, le polyèdre SM $S = A \oplus B$. (a) La SM du modèle Knot (genre élevé) et du modèle Sphere. (b) La SM du modèle Grate1 et du modèle TruncatedTetrahedron. Le chevauchement des dents est correctement géré. (c) La SM du modèle Grate2 (modèle conmplexe) et du modèle RhombicDodecahedron. (d) La SM du modèle Wrench et du modèle RhombiTruncatedIcosahedron. Le changement de genre est géré correctement. (e) La SM du modèle Bunny et du modèle Sphere. (f) La SM du modèle Dinausor et des modèles Cube/Sphere. L'algorithme NCC-CVMS gère les polyèdres de grande taille.*