Deux algorithmes d'intersection des surfaces de subdivision

S. Lanquetin, S. Foufou, H. Kheddouci, M. Neveu

LE2I, FRE CNRS 2309/Université de Bourgogne BP 47870
21 078 Dijon cedex

sandrine.lanquetin@u-bourgogne.fr

Résumé : *Le calcul des intersections de surfaces est un problème fondamental en modélisation. Toute opération Booléenne peut être vue comme un calcul d'intersection suivi d'une sélection des parties à conserver pour construire la surface de l'objet résultant. Dans cet article, nous nous intéressons aux calculs des intersections de surfaces de subdivision (surfaces générées par le schéma de subdivision de Loop). Nous présentons trois variantes d'algorithmes de calcul différent. La première variante calcule cette intersection après une classification des faces des objets en couples d'intersection et de non intersection. La seconde variante se base sur le 1-voisinage des faces en intersection. La troisième variante utilise la notion de graphe biparti.*

Mots-clés : Opérations booléennes, courbes d'intersection, surface de subdivision, principe de Loop.

1. Introduction

Les méthodes de génération de surfaces occupent une place très importante en Infographie et en Conception et Fabrication Assistée par Ordinateur (CFAO). La modélisation basée sur les surfaces de subdivision dispose de deux avantages principaux. Elle s'applique à des maillages de topologie arbitraire (comme la modélisation polygonale) et elle a un comportement local (comme la modélisation par les NURBS ou les B-Splines) puisqu'elle s'appuie uniquement sur un petit nombre de points de contrôle.
Les surfaces de subdivision sont maintenant largement utilisées. Le succès de ces surfaces provient de leur capacité à générer des surfaces lisses à partir de maillages initiaux arbitraires et leur implémentation relativement aisée grâce à leur concept simple. Elles sont définies par un maillage initial de type arbitraire et des règles de raffinement. Ces règles sont composées de règles géométriques déterminant les positions des nouveaux points de contrôle à partir des positions des anciens et de règles topologiques qui décrivent la procédure de raffinement de la connectivité du polyèdre de contrôle et de ce fait les propriétés de la surface. A partir d'un maillage polygonal, appelé réseau de contrôle, l'application répétée des règles de raffinement produit des nouveaux maillages polygonaux comprenant de plus en plus de faces. La suite de maillages ainsi constituée converge vers une surface lisse, appelée surface limite (par exemple, B-spline ou Box-spline), topologiquement similaire au réseau de contrôle initial. La Figure 1 montre un exemple de surface de subdivision. De la gauche vers la droite, la surface est de plus en plus lisse.

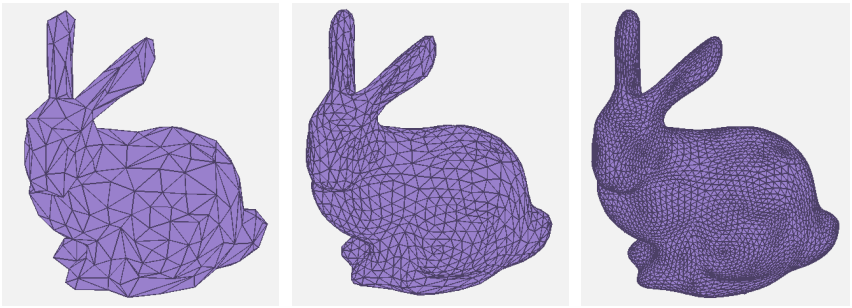

Fig. 1. Exemple de surface de subdivision.

Depuis l'introduction des surfaces de subdivision en 1978 par Catmull-Clark [Cat78] et Doo-Sabin [Doo78] de nombreux principes (ou schémas) de subdivision ont été proposés [Loo87, Zor00, Kob00, Vel00].
Le tableau 1 présente une synthèse des principes les plus connus. Pour chaque schéma, on peut constater le type de maillage sur lequel il peut s'appliquer (triangulaire, quadrilatéral…) et le type de la surface limite.

Schéma	Type de maillage	Type de surface limite
Doo-Sabin	polygonal	B-spline quadratique
Catmull-Clark	quadrilatéral	B-spline cubique
Loop	Triangulaire	B-spline triangulaire quartique

Tab. 1. Classification des schémas de subdivision les plus courants.

L'utilisation croissante de ces surfaces nécessite de reconstruire les outils préalablement connus pour d'autres types de surfaces ou de solides. Le calcul des opérations booléennes, par exemple, est fondamental pour la construction d'objets complexes à partir d'objets plus simples. Un objet CSG est généré par combinaison de plusieurs opérations booléennes (intersection, union, différence) entre des primitives élémentaires [Kri94]. Les primitives peuvent être des formes simples (cube, cylindre, tore…) ou des formes plus complexes construites à l'aide d'un ensemble de primitives simples ou généré par des surfaces plus compliquées. La Figure 2 montre un exemple d'opérations booléennes pouvant être effectuées sur deux objets simples.

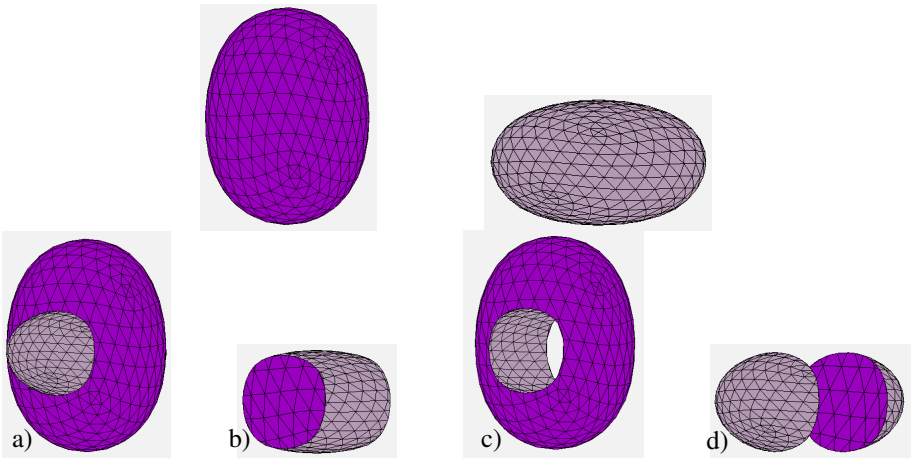

Fig. 2. Union (a), intersection (b) et différence ((c) et (d)) entre les deux surfaces.

De façon générale, une opération booléenne se déroule en deux étapes :

- Calcul des courbes d'intersection entre les surfaces des deux primitives impliquées dans l'opération : des points d'intersection sont trouvés, classés puis connectés pour approcher la courbe d'intersection.
- Conservation des parties des primitives nécessaire à la construction de la surface de l'objet résultant selon l'opération booléenne envisagée.

En modélisation géométrique, le calcul des intersections de surfaces est un problème récurrent et fondamental largement étudié pour les surfaces algébriques et paramétriques [Boe91, Pat93, Abd96]. Selon les algorithmes impliqués dans les différentes taches fondamentales, les méthodes de calcul d'intersections peuvent être classées parmi les quatre catégories principales suivantes. Les méthodes analytiques [Pat93, Cha87] lorsqu'une des surface est exprimée sous forme implicite et l'autre sous forme paramétrique. Les méthodes de discrétisation réduisent le problème d'intersection surface/surface à un ensemble de problèmes d'intersection courbe/surface [Bar87]. Les méthodes de suivi nécessitent de connaître au moins un point de la courbe d'intersection, appelé point de départ, pour générer une suite de points sur la courbe d'intersection en exploitant les propriétés géométriques locales des surfaces en intersection [Baj88]. Les méthodes de subdivision utilisent le raffinement des surfaces pour trouver une approximation polygonale des courbes d'intersection. La fiabilité de ces méthodes dépend du niveau de subdivision et des différents outils utilisés pour contrôler l'arbre de subdivision.

Dans cet article, nous nous intéressons particulièrement au calcul des intersections des surfaces de subdivision dans un contexte d'algèbre de solides modélisés par des surfaces de subdivision. Pour des raisons pratiques, nous ne considérons que les surfaces de subdivision générées par le principe de Loop. Ce principe s'appliquant uniquement sur des maillages triangulaires, il permet de travailler sur des faces triangulaires, et par conséquent planes. Nous présentons et nous comparons trois variantes d'un algorithme de calcul : la première variante calcule cette intersection après une classification des faces des objets en couples d'intersection et de non intersection. La seconde variante se base sur le 1-voisinage des faces en intersection. La troisième variante utilise la notion de graphe biparti. Pour appliquer les deux dernières variantes, il est nécessaire de connaître les courbes d'intersection au premier niveau. Elles seront donc calculées à l'aide de l'algorithme naturel d'intersection de maillages. La principale différence entre la version voisinage et la version graphe repose sur les ensembles de faces considérés. La première considère un ensemble de faces par objet alors que la seconde fait intervenir plusieurs sous ensemble de l'ensemble précédent par l'intermédiaire d'un graphe biparti.

2. Concepts de base.

La deuxième et la troisième variante de notre algorithme se basent sur les notions de 1-voisinage, de 2-voisinage et de graphe biparti. La Figure 3 illustre les termes de 1-voisinage et de 2-voisinage. Le 1-voisinage $\mathcal{V}_1(F)$ d'une face F contient cette face et toutes les faces voisines à cette face par un sommet, il est délimité par les petits pointillés : $\mathcal{V}_1(F) = \{F\} \cup \{F_i / \exists s \in S(F) \cap S(F_i)\}$ où $S(F)$ est l'ensemble des sommets de la face F. Le 2-voisinage $\mathcal{V}_2(F)$ est en fait le 1-voisinage de $\mathcal{V}_1(F)$, il est borné par les grands pointillés : $\mathcal{V}_2(F) = \{\mathcal{V}_1(F_i) / F_i \in \mathcal{V}_1(F)\}$. L'ensemble des faces obtenues par une subdivision du 1-voisinage de F est noté $\mathcal{W}_2(F)$, $\mathcal{W}_2(F) = \{F_i / F_i \in Loop(\mathcal{V}_1(F))\}$. $\mathcal{W}_1(F)$ est un sous-ensemble de faces de $\mathcal{W}_2(F)$ restreint aux faces ayant un sommet commun avec l'une des sous faces résultant de la subdivision de F, $\mathcal{W}_1(F) = \{F_i / F_i \in Loop(\mathcal{V}_1(F)) \,\&\, s \in S(Loop(F)) \cap S(F_i)\}$ où $S(Loop(F))$ est l'ensemble des sommets obtenus par la subdivision de F.

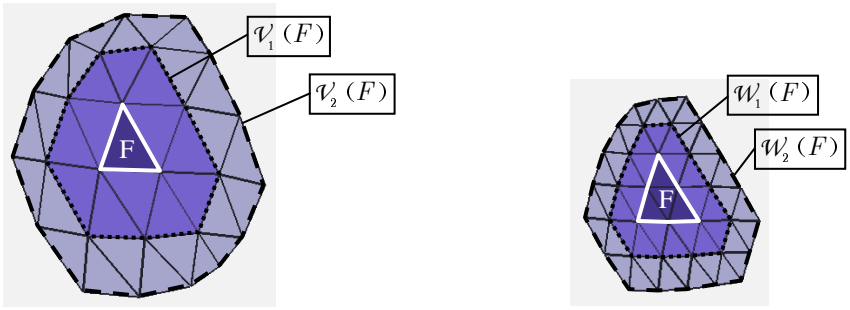

Fig. 3. 1-voisinage et 2-voisinage : $\mathcal{V}_1(F)$, $\mathcal{V}_2(F)$, $\mathcal{W}(F)$ et $\mathcal{W}_2(F)$.

Un graphe biparti est un graphe dans lequel :

- Les sommets sont répartis en deux groupes I_1 et I_2. Dans notre algorithme, ces groupes seront respectivement formés par les faces intersectantes de chaque surface.
- Chaque arête a une de ses extrémités dans chacun de ces groupes. Dans notre cas, une arête symbolisera la présence d'une intersection entre deux faces.
- Aucune arête ne peut relier deux sommets d'un même groupe.

3. La variante naturelle

Le calcul des courbes d'intersection s'effectue en plusieurs étapes. Tout d'abord, les faces des deux surfaces sont répertoriées en deux catégories : les faces en intersection et les autres. Seul les couples de faces en intersection sont considérés dans la suite de l'algorithme. L'intersection face/face est calculée pour obtenir les points d'intersection qui seront ensuite triés et reliés par des segments de manière à obtenir des approximations linéaires par morceaux des courbes d'intersection.

3.1. Détection des intersection entre deux faces

Cette étape préliminaire consiste à utiliser les tests de collision des boites englobantes des faces des deux maillages pour faire un premier filtrage qui éliminera les faces clairement disjointes de toute investigation future. Ensuite, les intersections de toutes les faces restantes vont être calculées. La complexité de l'algorithme devient en $O(m_1 \times n_1)$ avec m_1 et n_1 respectivement inférieurs aux nombre de faces m et n des surfaces S_1 et S_2.

3.2. Déterminer les points d'intersection entre deux faces

Plusieurs méthodes peuvent être envisagées pour calculer les points d'intersection. O'Brien et Manocha [Obr00] calculent l'intersection en effectuant l'intersection des plans porteurs des faces puis en prenant la restriction aux faces. Ils sont donc obligés de distinguer deux cas : le cas où les deux points d'intersection appartiennent aux segments d'une même face et le cas où ils sont portés par des segments de faces différentes (Figure 4).

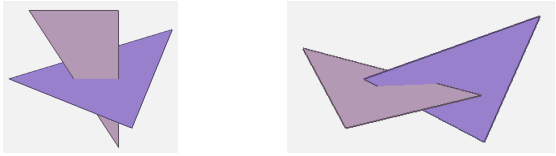

Fig. 4. Deux cas d'intersection face/face.

Une autre solution consiste à calculer les intersections de tous les segments d'une face avec l'autre face. L'intersection droite plan est tout d'abord calculée. L'équation de la droite étant écrite sous forme barycentrique, la restriction au segment s'effectue en vérifiant que la valeur du paramètre est dans l'intervalle $[0,1]$. Ensuite il ne reste plus qu'à vérifier l'appartenance du point d'intersection à la face, ce qui se fait aisément en comparant l'aire de la face avec la somme des aires des triangles formés par le point d'intersection et les sommets de la face. C'est cette seconde méthode que nous avons choisi d'implémenter.

3.3. Tri des points d'intersection et évaluation de la courbe polygonale d'intersection.

Le tri des points est facilement réalisé grâce à la structure du point d'intersection qui stocke les coordonnées du point, les faces de chaque objet à l'origine de ce point et l'arête qui porte ce point. Ainsi les extrémités des segments d'intersection correspondent aux points d'intersection contenant deux faces identiques. Ensuite, les segments sont reliés entre eux en utilisant la structure winged edge [Bau72].

4. La variante de voisinage.

Le calcul de l'intersection au niveau initial pour cet algorithme se fait à l'aide de l'algorithme naturel. Les faces en intersection des deux surfaces sont donc connues par la suite. La Figure 5 représente la courbe d'intersection et les faces en intersection au niveau initial.

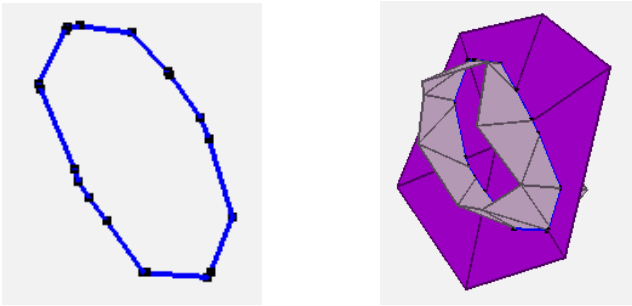

Fig. 5. Courbe d'intersection au premier niveau avec les faces en intersection.

Soit I_i l'ensemble des faces participant à l'intersection de la surface S_i. La première étape de cet algorithme consiste à récupérer le 1-voisinage $\mathcal{V}_1(I_i)$. Rappelons que ce 1-voisinage contient l'ensemble I_i ainsi que toutes les faces voisines aux faces de I_i par un sommet. Les ensembles $\mathcal{V}_1(I_1)$ et $\mathcal{V}_1(I_2)$ sont représentés par la Figure 6.a.

Puis les ensembles $\mathcal{V}_1(I_i)$ sont subdivisés en appliquant le principe de Loop. Lors de cette étape de raffinement, seul le 1-voisinage des ensembles $\mathcal{V}_1(I_i)$ est conservé. En effet, pour obtenir une subdivision correcte du 1-voisinage en entier, le 2-voisinage serait nécessaire. Sur la Figure 6.a, les faces obtenues par subdivision de I_i sont représentées en foncé et celles du 1-voisinage en clair sur la Figure 6.b.

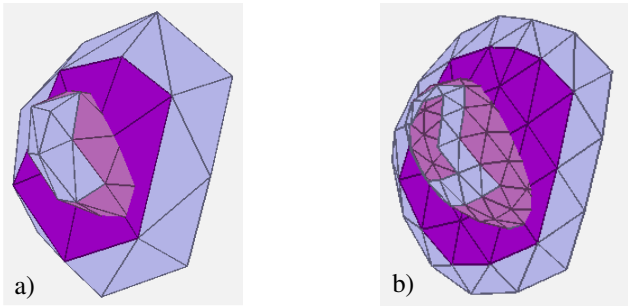

Fig. 6. Faces en intersection. a) 1-voisinage $\mathcal{V}_1(I_i)$. b) 1-voisinage de la subdivision $\mathcal{W}_1(I_i)$.

En réalité, on conserve tout de même le 2-voisinage $\mathcal{V}_2(I_i)$ de I_i afin de calculer correctement le 2-voisinage $\mathcal{W}_2(I_i)$ de l'ensemble des sous faces des faces I_i obtenu par la subdivision de Loop. En effet, dans certains cas, il est nécessaire d'avoir les faces de $\mathcal{W}_2(I_i)$ en mémoire pour récupérer le voisinage en entier au niveau suivant. Ensuite, l'algorithme naturel est à nouveau utilisé pour tester les intersections entre $\mathcal{W}_1(I_1)$ et $\mathcal{W}_1(I_2)$. Les courbes d'intersection obtenues au niveau suivant peuvent être tracées (Figure 7).

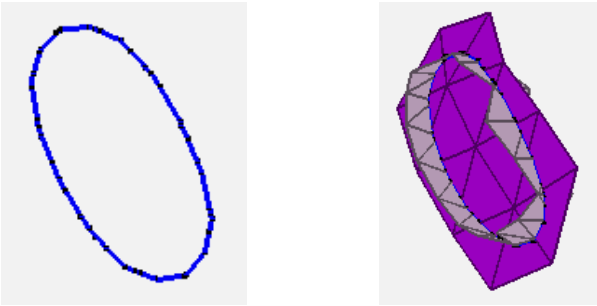

Fig. 7. Intersection au niveau 1. Gauche : la courbe obtenue. Droite : les faces participant à l'intersection.

Pour récupérer les courbes d'intersection au niveau de subdivision x donné, il suffit d'appliquer x fois ce processus. La Figure 8 montre les résultats obtenus au niveau 2. De gauche à droite, on a : les 1-voisinages $\mathcal{V}_1(I_i)$, les 1-voisinages de la subdivision $\mathcal{W}_1(I_i)$, les faces de $\mathcal{W}_1(I_i)$ en intersection et la courbe d'intersection.

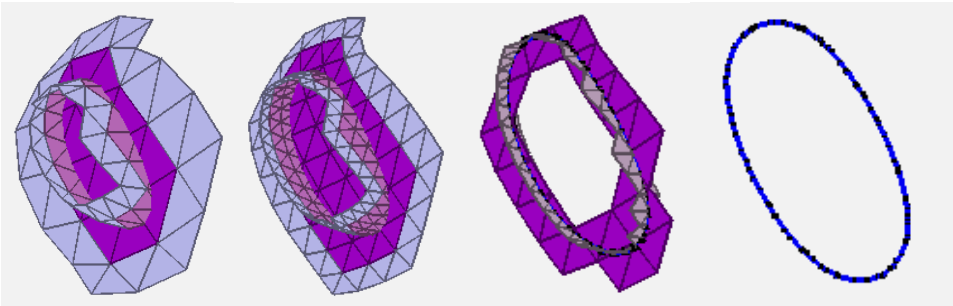

Fig. 8. Voisinage, subdivision et courbe d'intersection au niveau 2.

Cet algorithme réduit de manière significative le nombre de faces intervenant dans le calcul d'intersection, il est par conséquent beaucoup plus rapide que l'algorithme naturel.

5. Algorithme utilisant un graphe biparti

Cet algorithme repose sur l'utilisation d'un graphe biparti, il permet de réduire le nombre d'intersections à tester. Dans l'exemple présenté sur la Figure 9, on construit le graphe d'intersection d'une bande (40 faces) avec un lapin (694 faces).

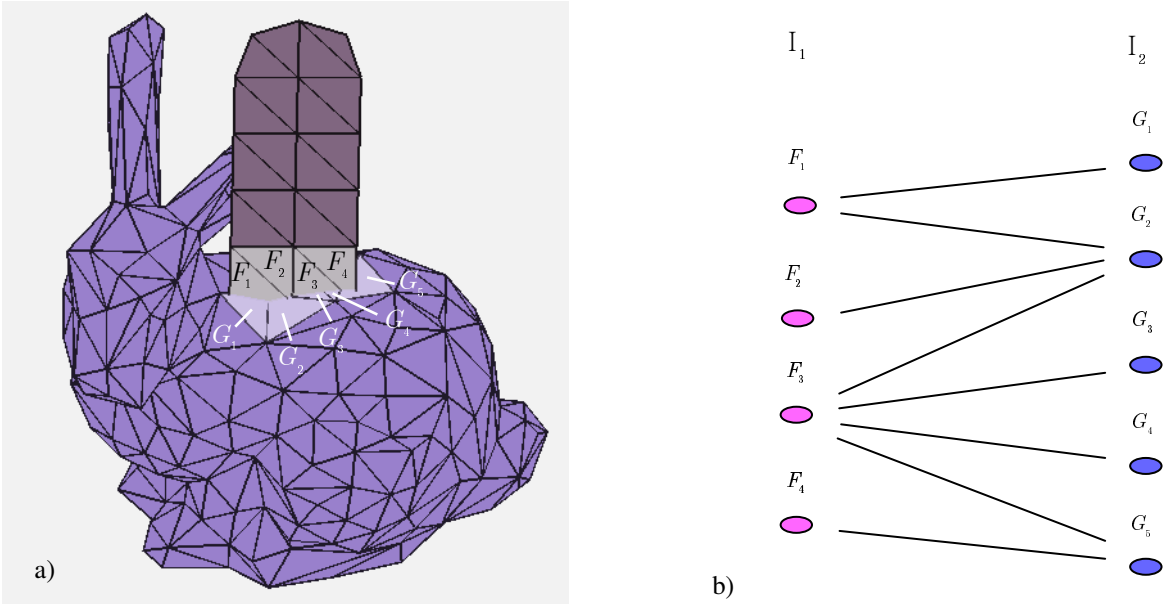

Fig. 9. Exemple de construction d'un graphe où F_i et G_i sont respectivement des éléments de I_1 et de I_2.

Les faces intersectantes de la bande sont appelées de gauche à droite F_1 à F_4 et celle du lapin G_1 à G_5 (Figure 9.a). Les nœuds de la première partie du graphe, disposés en colonne représentent les faces de l'objet 1 (la bande) et ceux de la seconde partie représentent les faces de l'objet 2 (le lapin). Les faces de l'objet 1 qui intersectent l'objet 2 sont reliées par des arêtes (Figure 9.b).
Une fois ce graphe constitué, les voisinages des faces des 2 objets sont ajoutés au graphe. L'intersection de chaque face de $\mathcal{V}_1(I_1)$ avec toutes les faces de $\mathcal{V}_1(I_2)$ n'est plus testée, l'idée de l'algorithme consiste à calculer seulement les intersections entre des sous-groupes de ces ensembles. En effet, à chaque face intersectante F_i de la surface S_1, le graphe d'intersection (graphe biparti) fait correspondre certaines faces G_i de la surface S_2 en intersection avec cette face, on va donc calculer uniquement les intersections entre $\mathcal{V}_1(F_i)$ et $\mathcal{V}_1(G_k)$ étant reliés par une arête.

L'entrée de cet algorithme est un graphe biparti $G^0(I_1^0, I_2^0, E^0)$ où I_1^0 et I_2^0 sont les ensembles de sommets du graphe (faces intersectantes) et E^0 l'ensemble des arêtes (couples de faces en intersection). Sa sortie est également un graphe biparti $G^1(I_1^1, I_2^1, E^1)$ où I_1^1 et I_2^1 sont les ensembles de sommets (faces intersectantes) et E^1 l'ensemble des arêtes. Ce graphe représente le graphe biparti au niveau suivant de subdivision (couples de faces en intersection).
L'algorithme opère en 4 étapes consécutives de la façon suivante :

1. Pour chaque nœud du graphe, les 1-voisinages des faces F_i de l'objet 1 et des faces G_k de l'objet 2 sont récupérés, ils sont respectivement notés $\mathcal{V}_1(F_i)$ et $\mathcal{V}_1(G_k)$.
2. Les $\mathcal{V}_1(F_i)$, $F_i \in I_1$ et $\mathcal{V}_1(G_k)$, $G_k \in I_2$ sont ensuite subdivisés partiellement de manière à conserver uniquement le 1-voisinage de la subdivision de F_i et de G_k et notés $\mathcal{W}_1(F_i)$ et de $\mathcal{W}_1(G_k)$.
3. L'algorithme naturel est utilisé pour déterminer les couples de faces intersectantes entre $\mathcal{W}_1(F_i)$ et $\mathcal{W}_1(G_k)$ (cf. section 3).
4. Le nouveau graphe biparti $G^1(I_1^1, I_2^1, E^1)$ est construit à l'aide de ces nouveaux couples de faces intersectantes de la même manière qu'à la Figure 9.

Ci-dessous, nous appliquons l'algorithme précédent sur un exemple simple. Pour réduire le nombre de sommets du graphe, cette fois on considère l'intersection du lapin (694 faces) avec une plus petite bande (10 faces).
Les Figures 10 à 13 illustrent successivement la construction du graphe biparti initial, l'étape 1 de récupération du voisinage, l'étape 2 de subdivision partielle, l'étape 3 de détermination des couples de faces en intersection et l'étape 4 de construction du nouveau graphe d'intersection.

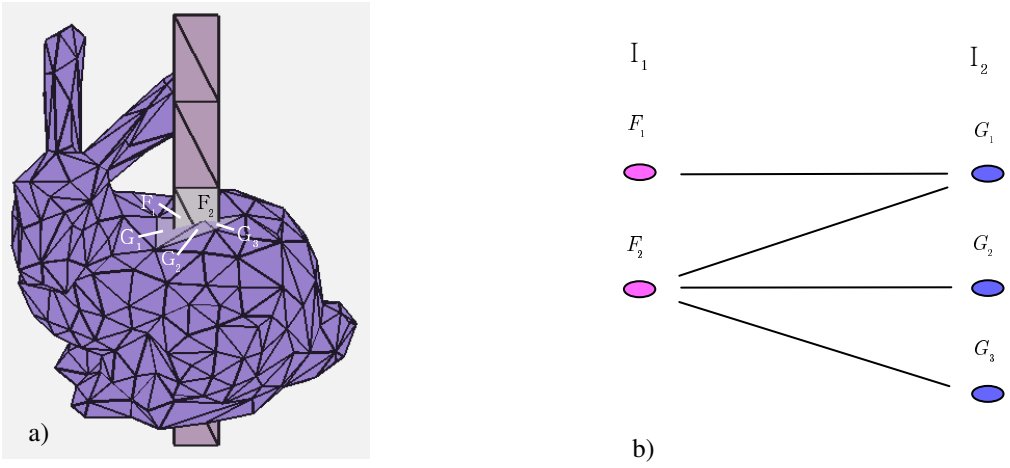

Fig. 10. Construction du graphe biparti initial

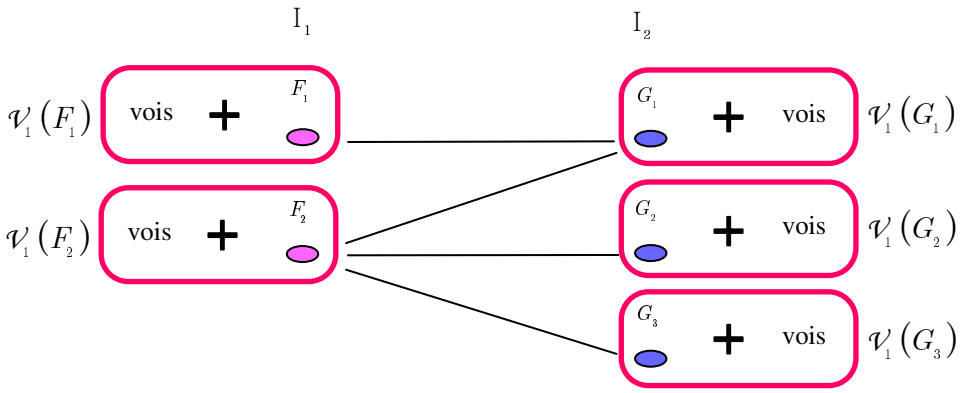

Fig. 11. Etape 1 : Récupération des 1-voisinages $\mathcal{V}_1(F_i)$ et $\mathcal{V}_1(G_k)$.

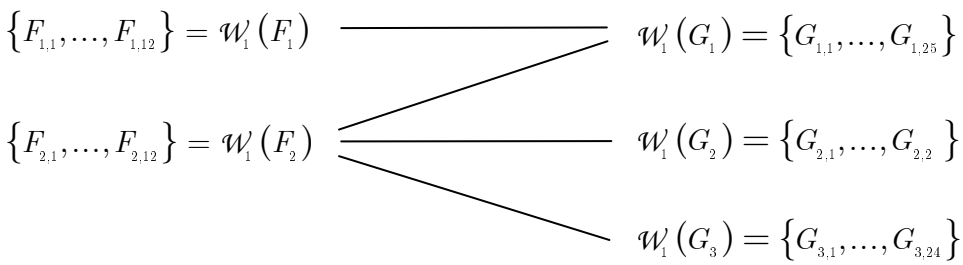

Fig. 12. Etape 2 : Obtention des $\mathcal{W}_1(F_i)$ et de $\mathcal{W}_1(G_k)$ par subdivision partielle de $\mathcal{V}_1(F_i)$ et $\mathcal{V}_1(G_k)$.

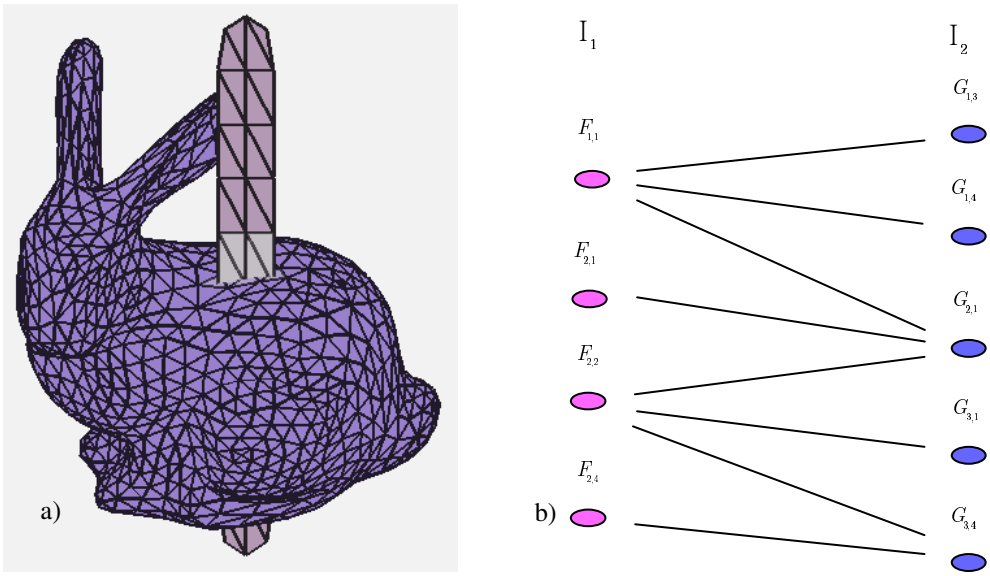

Fig. 13. Construction du nouveau graphe biparti à l'aide des couples de faces en intersection de $\mathcal{W}_1(F_i)$ et de $\mathcal{W}_1(G_k)$.

Cet algorithme est plus efficace puisqu'il se base sur un graphe biparti donnant un minimum de faces intersectantes à chaque subdivision. Actuellement, l'implémentation est en cours afin de confirmer cette affirmation. En effet, l'idée d'ordonner les sommets de I_1 et de I_2 est envisagée afin d'éviter de générer plusieurs fois les même faces en calculant les 1-voisinages de ces sommets.

6. Conclusion

Dans cet article, nous avons décrit trois algorithmes permettent de calculer les courbes d'intersection entre deux objets modélisés par des surfaces de subdivision. L'algorithme naturel est une variante peu optimisée qui peut

être intéressante à utiliser lorsque les surfaces des objets à intersecter ont un nombre réduit de faces. Les deux autres algorithmes proposés reposent sur les notions de 1-voisinage et de graphe biparti. Ils permettent de calculer les courbes d'intersection entre deux objets plus rapidement qu'avec l'algorithme naturel notamment lorsque le nombre de faces en présence est très élevé. Il reste maintenant à intégrer cette amélioration dans le cadre des opérations booléennes. L'implémentation de ces deux algorithmes est en cours. Une étude comparative sera envisagée par la suite avec d'autres algorithmes existants.

Références

[Abd96] K. Abdel-Malek, H. J. Yeh. "Determining intersection curves between surfaces of two solids". Computer Aided Design, vol. 28-6/7, pp 539-549, 1996.

[Baj88] C. L. Bajaj, C. M. Hoffmann, J. E. Hopcroft, R. E. Lynch. "Tracing surface intersections". Computer Aided Geometric Design, vol. 5, pp 285-307, 1988.

[Bar87] R. E. Barnhill, G. Farin, M. Jordan, B. R. Piper. "Surface/surface intersection". Computer Aided Geometric Design, vol. 4-3, pp 3-16, 1987.

[Bau72] Bruce G. Baumgart, "Winged edge polyhedron representation", Technical Report CS-TR-72-320, pp 5, 1972.

[Boe91] E. Boender. "A survey of intersection algorithms for curved surfaces". Computer & Graphics, vol. 15-1, pp 99-115, 1991.

[Cat78] E. Catmull, J. Clark. "Recursively generated B-spline surfaces on arbitrary topological meshes". Computer Aided Design, vol. 9-6, pp 350-355, 1978.

[Cha87] V. Chandru, B. S. Kochar. "Geometric Modeling: Algorithms and NEW Trends". Chapter Analytic Techniques for Geometric Intersection Problems, pp 305-318, SIAM, Philadelphia, PA, 1987.

[Doo78] D. Doo, M. Sabin. "Behaviour of recursive subdivision surfaces near extraordinary points". Computer Aided Design, vol. 9-6, pp 356-360, 1978.

[Kob00] L. Kobbelt. "Sqrt(3)-Subdivision". Computer Graphics Proceedings, Annual Conference Series, pp. 103-112, July 2000.

[Kri94] S. Krishnan, A. Narkhede, D. Manocha. "Boole: A System to Compute Boolean Combinations of Sculptured Solids". Technical Report, Department of Computer Science, University of North California, 1994.

[Loo87] C. Loop. "Smooth Subdivision Surfaces Based on Triangles". Master's thesis, University of Utah, Department of Mathematics, 1987.

[Obr00] D. A. O'Brien, D. Manocha. "Calculating Intersection Curve Approximations for Subdivision Surfaces", 2001. http://www.cs.unc.edu/~obrien/courses/comp258/project.html

[Pat93] N. M. Patrikalakis. "Surface-to-surface intersections". IEEE Computer Graphics & Applications, vol. 13-1, pp 89-95, January 1993.

[Vel00] L. Velho, D. Zorin. "4-8 Subdivision". Computer Aided Geometric Design, volume 18-5, pp 397-427, 2000.

[Zor00] D. Zorin. "Subdivision Zoo". SIGGRAPH 2000 Course Notes, Subdivision for Modeling and Animation, Chap. 4, pp 65-98, 2000.