Accélération de la détection de collisions entre corps rigides et déformables

P. Meseure, L. Hilde, C. Chaillou

LIFL - Université de Lille 1 Bât M3
59655 Villeneuve d'Ascq CEDEX FRANCE
{meseure,chaillou}@lifl.fr

RÉSUME :

Nous proposons dans cet article un procédé permettant de détecter en temps réel les collisions impliquant des corps déformables. Nous nous plaçons dans l'hypothèse d'objets définis comme un ensemble de triangles, ce qui est justifié car ces modèles forment la base des environnements virtuels actuels. Le calcul exact des collisions avec des corps déformables entraîne un coût de calcul très lourd et une complexité quadratique en fonction du nombre de facettes des objets. Nous étudions comment employer les algorithmes d'accélération de la détection dans un contexte d'objets déformables. Nous nous focalisons sur deux algorithmes : le premier à subdivision spatiale, puis le second à volumes englobants, adapté de l'algorithme des OBB-Trees et qui fournit des temps de détection trois fois plus rapides que le premier.

1. INTRODUCTION

Par définition, une collision survient lorsqu'au même temps *t*, deux objets occupent la même position de l'espace. La recherche des collisions implique en général des calculs d'intersection assez lourds, si bien que des études ont été menées pour accélérer cette détection. Cependant, elles ne se sont intéressées qu'à des environnements constitués de corps rigides. Il est vrai que ces environnements sont suffisants la plupart du temps. Pourtant, ces dernières années de nouvelles applications nécessitant l'introduction de corps déformables apparaissent. La simulation chirurgicale en fait partie et c'est dans ce contexte que se sont déroulés nos travaux. Notre détection de collisions est testée dans l'environnement d'un simulateur d'interventions cœlioscopiques en gynécologie [1]. La partie logicielle de ce simulateur doit intégrer un moteur qui calcule le comportement dynamique de corps déformables [2].

Dans cet article, nous nous intéressons donc plus particulièrement à la détection de collisions impliquant des corps déformables. Aucune supposition n'est faite sur leurs déformations au cours du temps. Certains corps sont immobiles, tandis que d'autres peuvent être déplacés et/ou déformés. Par contre, nous faisons l'hypothèse d'objets modélisés sous forme polyédrique, c'est à dire constitués d'un ensemble de polygones que nous supposerons triangulaires. Cette hypothèse est peu restrictive puisque la plupart des environnements virtuels sont modélisés de cette manière et que les afficheurs nécessitent que tout objet soit converti en polygones pour un affichage temps-réel.

Dans la suite, nous exposons dans un premier temps les contraintes liées aux corps déformables, puis nous nous intéressons dans les parties suivantes à l'adaptation de deux méthodes d'accélération. La première est classique et repose sur une subdivision régulière de l'espace, tandis que la suivante est basée sur des boîtes englobantes hiérarchi-

ques. Nous comparons dans la dernière partie les résultats obtenus par ces deux méthodes.

2. PROBLÉMATIQUE DES CORPS DÉFORMABLES

Nous n'allons pas détailler comment se calcule une collision exacte et fine avec un corps déformable car cela a déjà été décrit dans [3] et [4]. En résumé, les différents cas de collisions entre corps déformables ou entre corps rigides et déformables sont résolus par une équation linéaire de degré au plus 3, donc en temps constant. Comme chaque primitive du premier corps est testée avec celle du second corps, la complexité de l'ensemble de la détection est quadratique. Des mesures ont été effectuées sur diverses situations impliquant des corps déformables constitués de quelques dizaines à quelques centaines de nœuds. Elles montrent que la détection des collisions représente entre 90% et presque 100% du temps de calcul de la simulation. Il est donc nécessaire d'employer des techniques d'accélération.

Ces dernières cherchent non pas à détecter les collisions, mais à trouver le plus rapidement possible les non-collisions : en quelques tests simples il s'agit de pouvoir écarter un maximum de cas, afin de n'effectuer les calculs de collisions exacts que sur les paires de triangles qui ont une grande probabilité d'entrer effectivement en collision. Beaucoup de méthodes proposées ne sont pas ou difficilement compatibles avec les corps déformables. En effet, nombreuses sont celles qui supposent soit que le corps a une géométrie invariante soit qu'il est complètement caractérisé par sa position et son orientation. A chaque pas de la simulation, il faut en général recalculer la structure accélératrice pour l'adapter à la nouvelle forme de l'objet et une grande partie du gain de l'accélération est perdue dans cette mise à jour. Par exemple, les méthodes à base de volumes englobants exigent un recalcul de ces volumes, qui peut être coûteux et pire encore si la structure est hiérarchique. Plusieurs méthodes font des hypothèses sur la convexité de la forme polyédrique des corps : si lors des déformations, le corps déformable n'était plus convexe, il faudrait en temps réel le découper en polyèdres convexes.

Nous restreignons donc notre étude à deux types de méthodes d'accélération : les algorithmes qui ne demandent aucune adaptation aux corps déformables et ceux dont les modifications n'entraînent pas de pertes de temps trop importantes. Dans les deux sections suivantes, nous étudions successivement l'utilisation d'une subdivision de l'espace puis nous proposons une adaptation de l'algorithme très performant des OBB-Trees [5].

3. ACCÉLÉRATION PAR VOXELS

Nous cherchons à éviter les calculs de collisions entre des objets qui ne sont pas situés dans la même zone de l'espace. Pour cela, nous divisons l'espace en une grille régulière de voxels. A chaque voxel est associée une liste de facettes. Lors de la simulation, l'ensemble des objets, rigides ou déformables est parcouru et leurs facettes sont placées dans la grille de voxels. Les facettes sont disposées dans la liste associée à chaque voxel par lequel elles passent. Une fois toutes les facettes disposées dans la grille, tous les voxels sont analysés. Si plusieurs facettes sont présentes dans le même voxel, il y a collision possible entre ces facettes, ce qui est testé avec les routines de détection exacte.

Les sources de ralentissement de cet algorithme sont tout d'abord le parcours de

l'ensemble des voxels de l'espace pour détecter les collisions. En outre, les listes associées aux voxels doivent être vidées à chaque nouveau pas de simulation. Cependant, une optimisation simple consiste à remplir une fois pour toutes cette grille avec les objets immobiles. Une autre optimisation permet de s'affranchir de vider les voxels, en employant un mécanisme de datation de la liste (tout liste ancienne est considérée comme vide).

Dans notre contexte, le point fort de cet algorithme est de traiter de façon identique les objets rigides mobiles et les corps déformables. Seuls les objets considérés comme toujours immobiles bénéficient d'un traitement particulier. D'un autre côté, un gros inconvénient de cette méthode est la nécessité de choisir la précision de la subdivision : en effet, si la subdivision est trop grossière, de nombreuses facettes sont réparties dans les mêmes voxels, ce qui engendre généralement de nombreux calculs exacts de collisions superflus. Au contraire, si la grille est trop dense, les facettes doivent être réparties dans plusieurs voxels à la fois : le taux de duplication des facettes augmente, ces facettes sont donc traitées plus souvent. Il faut généralement trouver un compromis de façon empirique. Une solution simple pour la répartition est de prendre des tailles de voxels assez proches des tailles des facettes, mais elle n'est pas forcément optimale. Cette méthode a été implantée et comparée à celle que nous décrivons dans la suite.

4. L'ALGORITHME DES OBB-TREES

Les techniques d'accélération de la détection des collisions basées sur les volumes englobants sont parmi les plus employées. Parmi ces dernières, les méthodes hiérarchiques sont particulièrement intéressantes, puisqu'elles permettent à la fois de gérer simplement la détection des non-collisions, à gros grains, entre objets, et de déterminer les couples de facettes en collision. La méthode des OBB-Trees (Oriented Bounding Box Trees) [5] est connue comme l'une des plus performantes. Cette méthode offrant des performances très intéressantes pour les corps rigides, il paraît judicieux de chercher à l'exploiter pour des corps déformables.

4.1. PRINCIPE DE LA DÉTECTION

Dans la méthode OBBT, la structure accélératrice est un arbre binaire de boîtes englobantes associé à chaque objet. Chaque boîte possède une orientation indépendante de celle des autres. Le test de collisions revient à un parcours de deux arbres en parallèle, de la racine vers les feuilles. Lorsque les deux boîtes associées aux nœuds des deux arbres sont en collision, la recherche continue en descendant d'un niveau dans l'arborescence de la boîte la plus grande. Le parcours d'une branche s'arrête soit lorsqu'aucune collision n'est possible soit lorsque les feuilles ont été atteintes dans les deux arbres.

Le test de non-collision entre boîtes englobantes orientées revient à trouver un plan séparateur tel que les projections des boîtes englobantes sur l'axe orthogonal au plan ne s'intersectent pas. Ce plan est à trouver parmi 15 possibilités. Ces plans n'étant pas quelconques, les auteurs montrent que le calcul de la projection est simplifié et plus rapide.

Ces tests nécessitent deux cents opérations dans le pire des cas (boîtes en intersection). Mais, en moyenne, la non-interpénétration est détectée en parcourant la moitié des axes. Il faut donc en général cent opérations contre cent quarante-quatre calculs d'intersection droite/plan.

4.2. CONSTRUCTION D'UNE OBB

La méthode proposée repose sur des arguments statistiques. Le but est d'essayer de construire une boîte qui reste proche des dimensions de l'objet et qui aurait son orientation. L'algorithme de calcul de telles boîtes prend en compte la statistique du 1er et 2ème ordre, i.e. le point moyen μ et la matrice de covariance C. Pour une liste de n triangles, soient p^i, q^i et r^i les sommets du $i^{ème}$ triangle. les expressions de μ et de C sont alors :

$$\mu = \frac{1}{3n}\sum_{i=1}^{n}\left(p^i + q^i + r^i\right) \qquad (1)$$

$$C_{jk} = \frac{1}{3n}\sum_{i=1}^{n}\left(\bar{p}_j^i\bar{p}_k^i + \bar{q}_j^i\bar{q}_k^i + \bar{r}_j^i\bar{r}_k^i\right) \text{ pour } 1 \le i, j \le 3 \qquad (2)$$

où $\bar{p}^i = p^i - \mu$, $\bar{q}^i = q^i - \mu$ et $\bar{r}^i = r - \mu$. C est donc une matrice 3x3 symétrique. Cette matrice est diagonalisée en ayant l'assurance que les vecteurs propres sont mutuellement orthogonaux : ces derniers sont choisis comme axes de la boîte. Les sommets extrêmes le long de chaque axe permettent de déterminer les dimensions du parallélépipède rectangle recherché.

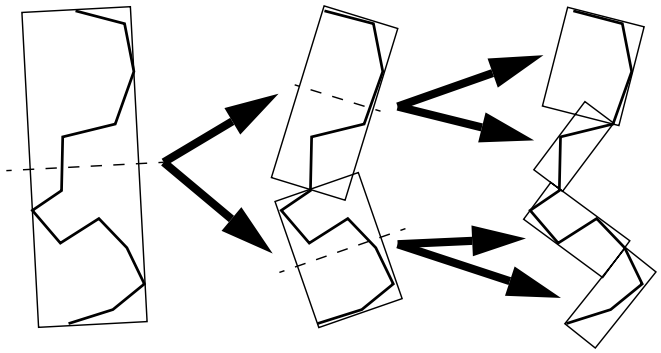

Figure 1. Construction de la hiérarchie des OBB

4.3. CONSTRUCTION DE L'ARBRE

L'arbre est construit à partir de la racine, en calculant une boîte qui englobe tout l'objet c'est à dire tous les triangles qui le constituent. Cet ensemble est ensuite divisé en deux, par un plan séparateur orthogonal à la plus grande dimension de la boîte. Chaque sous-ensemble sert à construire deux boîtes filles. Il suffit ensuite de réitérer le procédé de création de boîte englobante et de subdivision jusqu'à n'obtenir qu'un triangle par feuille. Afin que l'arbre soit équilibré, le plan séparateur est positionné de telle sorte que les deux boîtes filles contiennent à peu près le même nombre de triangles. Le début de construction de l'arbre est donné sur la *figure 1*.

5. ADAPTATION DES OBB-TREES AUX CORPS DÉFORMABLES

La méthode OBB-Tree exposée précédemment est cantonnée aux objets rigides. S'il faut prendre en compte des corps déformables, chaque boîte englobante doit être complètement recalculée (dimension, position et axes) à chaque pas. De plus, c'est

l'arbre tout entier qui doit être reconstruit en fonction de la nouvelle géométrie. Le coût de ces opérations est de l'ordre de plusieurs secondes de calcul.

Une première adaptation possible consiste à fabriquer l'arborescence une fois pour toute par la méthode originale. Cette arborescence reste inchangée tout au long de la simulation. Lors de la collision c'est toujours le test entre OBB qui est employé. A chaque pas de la simulation, les OBB sont recalculées en fonction des déformations du corps. Pour cela, en partant des feuilles et en remontant dans l'arbre, les OBB de chaque nœud sont mises à jour en fonction des OBB filles. Deux méthodes ont été exploitées [6]. Une OBB englobant les OBB des boîtes filles peut être construite. Cette méthode présente le défaut de générer des boîtes de dimensions trop grandes (voir *figure 2*). Il est également possible de se baser sur les statistiques μ et C qui ont servi à calculer les OBB filles, afin de trouver les statistiques de la boîte mère. Cependant, les études ont montré que dans ce cas, le calcul des vecteurs propres de la matrice C servant d'axes pour la boîte donne des temps de reconstruction prohibitifs (>100 ms).

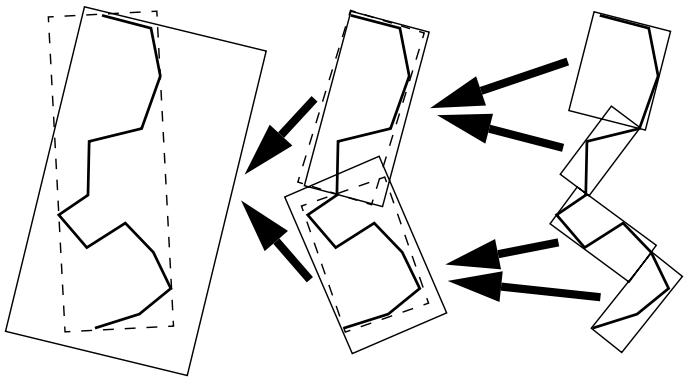

Figure 2. Reconstruction de la hiérarchie des OBB par calcul de boîte englobante de boîtes englobantes

6. UTILISATION D'OBB ET D'AAB

6.1. AVANTAGES DES AAB

La solution précédente pose le problème d'une mise à jour trop lourde de la hiérarchie lors de déformations de l'objet. Cette lourdeur est en grande partie causée par la recherche d'un nouveau repère pour chaque boîte de l'arborescence. Pour supprimer ce calcul, il faut travailler avec des boîtes exprimées dans le même repère. Le choix de ce repère est libre mais doit être fixé une fois pour toutes. A première vue, l'idéal est de minimiser la taille des boîtes, et de choisir par exemple le repère local de l'objet. Cependant, en pratique, ce choix n'est pas forcément intéressant à cause de la nature même du corps déformable. En effet, des boîtes optimales peuvent ne plus l'être après déformations. Il n'y a donc pas a priori de base particulière permettant de construire des boîtes plus petites. Il est donc plus judicieux de choisir le repère nécessitant le moins de calculs : comme tous les points sont exprimés dans le repère global, les boîtes sont donc construites dans cette base. De plus, le choix de ce repère permet des tests de collisions plus rapides.

Une hiérarchie d'AAB (Axis-Aligned Boxes) est donc choisie. Par rapport aux OBB, les AAB présentent le désavantage de moins respecter la géométrie initiale de

l'objet entouré. Lors de la simulation, des fausses collisions supplémentaires sont donc détectées et impliquent de devoir parfois descendre un peu plus dans les arbres pour trouver la non-collision. Cependant, cela n'entraîne pas forcément de pertes de temps [7]. En effet, dans la méthode OBB-Tree, l'usage de boîtes orientées oblige à calculer à chaque descente dans l'arbre le positionnement des boîte filles par rapport à la boîte mère ce qui entraîne des calculs de changement de repère. Les AAB étant toutes définies dans le même repère, il suffit d'un seul calcul au début de la descente.

6.2. CONSTRUCTION DE LA STRUCTURE D'ACCÉLÉRATION

La méthode employée est analogue à la méthode des OBB-Trees. Nous calculons, avant la simulation, une hiérarchie de boîtes alignées sur les axes du repère local de l'objet déformable au repos. Comme pour les OBB, une boîte englobante de l'ensemble des facettes est fabriquée. Cette boîte est ensuite découpée le long de son plus grand axe, afin de former deux listes disjointes de facettes qui seront attribuées aux fils droit et gauche. Si des facettes coupent le plan séparateur, elles sont mises dans le fils le moins chargé, afin d'équilibrer l'arbre. Ces boîtes ne servent qu'à la construction et non à la détection et ne sont pas mémorisées.

6.3. MISE À JOUR DE LA STRUCTURE ACCÉLÉRATRICE

La hiérarchie précédemment construite est gardée telle quelle durant toute la simulation. Lorsque le corps se déforme, l'arbre est mis à jour à partir des feuilles. Celles-ci sont traitées en calculant l'AAB du triangle contenu dans la feuille. Les AAB associées aux nœuds sont particulièrement simples à mettre à jour car il est très facile de calculer la boîte englobante mère de deux boîtes englobantes filles. Si les boîtes filles sont définies respectivement par les données $Xmin_1$ et $Xmax_1$ d'une part et $Xmin_2$ et $Xmax_2$ d'autre part, alors la boîte mère est définie par $Xmin=min(Xmin_1,Xmin_2)$ et $Xmax=max(Xmax_1,Xmax_2)$. Ceci est valable pour les trois axes. La mise à jour d'une boîte englobante en fonction de ses filles ne nécessitent donc que 6 tests et 6 affectations. La *figure 3* résume le procédé de mise à jour.

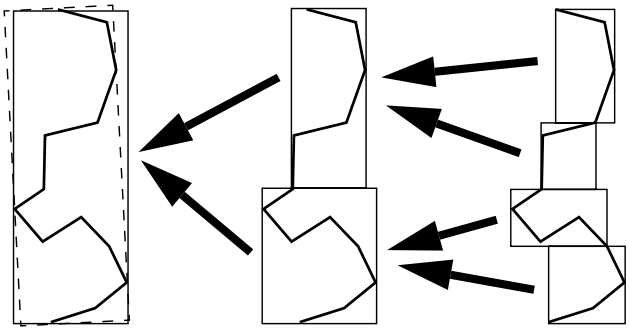

Figure 3. Mise à jour de la hiérarchie d'AAB

6.4. DÉTECTION DES COLLISIONS

Les AAB-Trees sont associées uniquement aux corps déformables ; les corps rigides continuent d'être englobés dans des structures OBB-Trees. Entre deux corps rigides,

la méthode OBB-Tree est donc toujours employée, avec les performances qui sont les siennes. Entre deux corps déformables, le parcours des deux arbres se base sur le même principe, mais nous employons un test entre boîtes AAB, puisque les hiérarchies de tous les corps déformables sont exprimées dans le même repère. Le test de non-collision entre corps AAB est simple puisqu'il revient à trouver une condition du type $(Xmin_1 > Xmax_2)$. Les autres conditions s'obtiennent en inversant le rôle des deux boîtes ou en testant un autre axe. Ce test revient alors à 6 comparaisons, dans le pire des cas.

Pour la collision d'un corps rigide et d'un corps déformable, il faut détecter les non-collisions entre des arbres d'OBB et d'AAB. Les arbres sont parcourus avec la même méthode. Les boîtes des deux arbres n'étant pas exprimées dans le même repère, nous employons le test OBB pour les séparer. C'est donc globalement l'algorithme des OBB-Trees qui est appliqué, mais sans les calculs de changement de repère entre les boîtes mère et filles de l'arbre AAB.

7. RÉSULTATS/DISCUSSION

Les résultats préliminaires montrent que les deux algorithmes d'accélération de la détection des collisions offrent des performances très intéressantes. L'environnement de la *figure 4* comprend un objet déformable de 512 faces et plusieurs objets rigides dont l'ensemble fait 3592 faces. Sans accélération, la simulation tourne à 0,1Hz. La fréquence passe à 16,5Hz avec les voxels et à 55Hz avec la méthode OBB/AAB. Ainsi, cette méthode offre un gain de 3 par rapport aux voxels et un gain de 550 par rapport à la solution non accélérée. Alors que pratiquement 100% du calcul était employé dans les collisions, avec la méthode OBB/AAB, la détection de collisions ne représente plus que 77% du calcul total de la simulation. Les tests ont été effectués sur un processeur R10000 cadencé à 195Mhz. Il est important de signaler que dans ces tests seul un objet rigide de 64 faces était considéré comme mobile, les autres objets rigides étant immobiles. Ces caractéristiques ont été exploitées dans la méthode voxels et sont inutiles pour la méthode OBB/AAB. Le gain de 3 est donc un gain minimal, en pratique l'accélération est bien supérieure. La méthode reste performante en augmentant le nombre de facettes : avec plusieurs corps déformables simulés (mais sans tenir compte des collisions entre eux), la fréquence passe à 30Hz pour 4x512 faces.

Par ailleurs, la méthode OBB/AAB offre de nombreux avantages sur la méthode voxels. Elle tient complètement compte de la nature rigide ou déformable des objets. Elle ne nécessite pas de paramétrisation, contrairement à la méthode voxel qui demande de déterminer au préalable la précision de la grille, en fonction de l'environnement simulé. Elle est enfin beaucoup plus résistante à la prise en compte du mouvement. En effet, ces deux méthodes considèrent qu'une collision est détectée lorsqu'une intersection à un instant donné est calculée. Or, il peut y avoir eu une collision, sans que les objets s'intersectent lors du test. C'est le cas lorsque par exemple deux corps sont passées l'un au travers de l'autre. De façon plus formelle, si les tests de collisions sont effectués tous les Δt et que les objets ont une vitesse maximale de $Vmax$, alors des collisions d'une distance d'au plus $\Delta l = Vmax * \Delta t$ ne seront pas détectées. Pour détecter toutes les collisions, une solution est de majorer tout déplacement par cette distance. Ainsi, pour les boîtes englobantes, il suffit de majorer la taille des boîtes en ajoutant sur chaque côté une marge de sécurité de longueur Δl. Pour la méthode à subdivision spatiale, cela revient à devoir

répartir les facettes dans les voxels voisins si les facettes sont à une distance inférieure à Δl de ces voxels. Cette solution a le défaut d'augmenter le taux de duplication des facettes.

En conclusion, nous avons montré qu'il était possible de bénéficier de la rapidité de la bibliothèque OBB-Tree même dans un contexte de corps déformables. Les résultats obtenus montrent que le traitement des collisions impliquant des corps déformables est possible en temps réel.

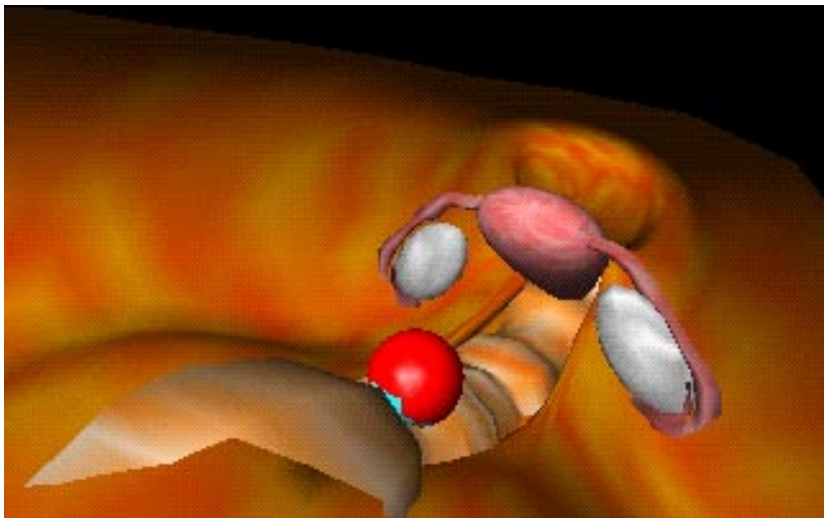

Figure 4. Copie d'écran dans l'environnement du simulateur.
La sphère est déformable et entre en collision avec le rectum,
les facettes en collisions sont colorées différemment.

RÉFÉRENCES

[1] Jambon, A.C., Dubois, P., et Karpf, S., "A Low-Cost Training Simulator for Initial Formation in Gynecologic Laparoscopy" *Proceedings of CVRMed'97*, Grenoble, France, 19-22 mars 1997, pp 347-356.

[2] Meseure, P., et Chaillou, C., "Deformable Body Simulation with adaptive subdivision and cuttings" *WSCG'97*, Plzen, 10-14 février 1997, pp 361-370.

[3] Meseure, P., et Chaillou, C., "Détection de collisions entre modèles polyédriques : quelques propositions" *Cinquième séminaire du groupe de travail "Animation et Simulation"*, Reims, 6-7 mars 1997, pp 91-100.

[4] Provot, X., "Animation Réaliste de Vêtements" *Cinquième séminaire du groupe de travail "Animation et Simulation"*, Reims, France, 6-7 mars 1997, pp 75-89.

[5] Gottschalk, S., Lin, M.C., Manocha, D., "OBBTree: a Hierarchical Structure for Rapid Interference Detection" *SIGGRAPH'96 Conference Proceedings, Computer Graphics annual conference series*, New Orleans, 4-9 Août 1996, pp 171-180, *http://www.cs.unc.edu/~geom/OBB/OBBT.html*.

[6] Hilde, L., *Accélération de la Détection de Collisions : Prise en Compte des Corps Déformables*, Mémoire de DEA d'informatique, Université de Lille 1, juillet 1997.

[7] Van Den Bergen, G., "Solid: Sophisticated Library for Interference Detection", URL : *http://www.win.tue.nl/cs/tt/gino/solid/*.