Détection de collisions entre modèles polyédriques : quelques propositions

P. Meseure, C. Chaillou

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

RESUME :
Cet article présente des améliorations des méthodes classiques de calcul des interactions entre objets polyédriques. Le calcul des interactions est découpé en deux étapes. La première consiste à éliminer les couples d'objets n'entrant pas en interaction pour fournir à l'étape suivante des couples d'objets ou de primitives d'objets entrant de façon probable en collision. La seconde étape est la détection proprement dite, c'est à dire la détermination du lieu et de l'instant de la collision lorsqu'elle a lieu. Les améliorations proposées sont la prise en compte des corps déformables dans la phase de détection, le calcul à la fois des collisions entre sommets et faces mais aussi entre arêtes et enfin une proposition d'accélération par boîtes englobantes qui tient compte de la trajectoire des objets entre deux pas de temps.
MOTS-CLES : Détection de collisions, Interactions, Réalité Virtuelle, Animation

1. Introduction

La détection de collisions, sans être un thème véritablement nouveau, connaît un regain d'intérêt ces dernières années. Elle consiste à comparer les positions des objets d'un environnement dynamique afin de déterminer si, à un instant donné, ces objets s'interpénètrent. A l'origine, le problème a d'abord été posé par les roboticiens pour des calculs de trajectoires de robots dans des lieux avec obstacles. Le problème revient alors à inspecter les trajectoires possibles et à vérifier que le robot peut emprunter réellement ces trajectoires. Depuis, la détection des collisions est devenu un problème central en synthèse d'images : elle est employée en animation et dans les environnements virtuels. Actuellement, la puissance de calcul nécessaire à une détection fine des interpénétrations empêche son emploi dans des contextes temps-réel. Or, cette étape est cruciale pour le réalisme. C'est en effet la détection des collisions qui empêche de passer au travers des murs, et au contraire permet de prendre un objet, le poser sur un support ou tout simplement ouvrir une porte. Par exemple, dans les spécifications de VRML 2.0 [Vrml96], les objets sont englobés par des volumes et lorsqu'une collision est détectée, on empêche l'utilisateur de se déplacer plus loin ou au contraire on fait en sorte qu'il glisse le long d'un plan... Ces solutions reviennent à considérer que la présence de l'utilisateur dans le volume implique l'utilisation immédiate. Si dans la majeure partie des cas, le fait de passer la main au voisinage d'un téléphone virtuel signifie effectivement que l'on désire contacter une personne, le fait de longer une porte ne signifie pas forcément qu'on veuille l'ouvrir. De même, le fait de passer à côté d'une télévision ne signifie pas forcément que on veuille l'allumer. Pire, si aucune collision réelle n'est détectée, il est probable que l'utilisateur passera la main au travers du téléphone, n'ouvrira pas la porte pour passer d'une pièce à une autre, et enfin pourra se tenir sans aucun problème dans la télévision.

Ces quelques anecdotes montrent que la détection des collisions est un problème crucial pour le réalisme des simulations. L'apport de la détection est immédiat : une fois la collision détectée, on peut avertir l'utilisateur par un signal sonore ou visuel ou mieux encore, avoir recours à un système à retour d'effort. Dans cet article, nous nous proposons de passer en revue les diverses techniques de détection de collisions. Nous définissons dans un premier temps comment les collisions peuvent être appréhendées et les diverses phases de la détection : à cause de son coût en temps de calcul, la détection est en général accompagnée d'une phase d'accélération. Ensuite, nous dressons un état de l'art à la fois des algorithmes de détection fine et d'accélération de la détection. Cette étude montre les points faibles des algorithmes existants et nous ferons quelques propositions pour les améliorer dans le cas des modèles polygonaux pour des contextes comme les mondes virtuels et l'animation.

2. Définition de la détection de collisions

Une collision entre deux objets est détectée s'il existe un instant où les deux objets ont une intersection non vide. [Boy79] considère que l'on peut soit tester les interpénétrations à des instants échantillonnés, soit au contraire tester les interpénétration le long de la trajectoire des objets. La première solution repose sur une détection des intersections (en 3D) à des instants discrets pour ensuite chercher par dichotomie l'instant exact où s'est produit la collision (voir *figure 1*). Cette méthode présente une limite importante : dans certains cas, des collisions ne sont pas détectées. Si deux objets sont très rapides, ils peuvent passer l'un au travers de l'autre entre deux instants où est effectué le test (voir *figure 2*). Une solution à ce problème est de considérer des intersections de trajectoires. Une première méthode consiste à considérer le volume balayé par les objets au cours du temps et de calculer les intersections entre ces volumes. Cependant, on peut avoir intersection des volumes sans avoir de réelles interpénétrations des objets (voir *figure 3*). Cette méthode présente donc le désavantage de détecter des collisions superflues.

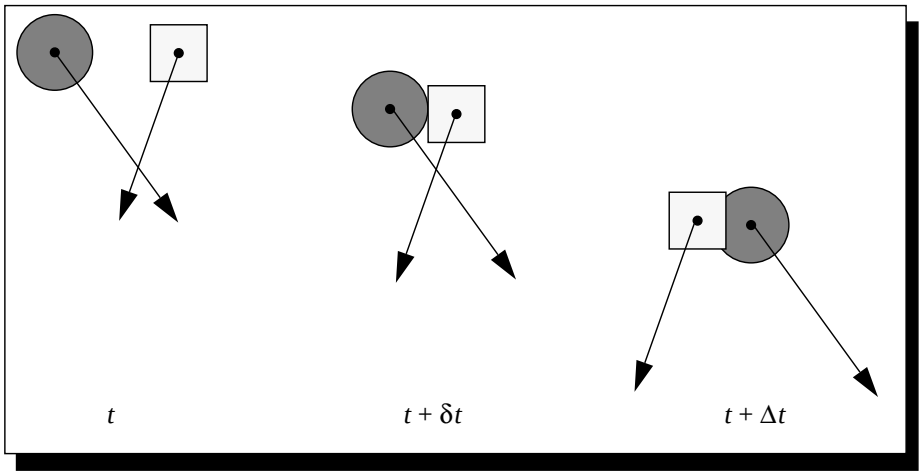

Figure 1. Détection de collisions par interpénétration

Pour éviter ces calculs inutiles, on peut se placer en dimension 4 c'est à dire considérer le temps comme une variable d'espace supplémentaire. On se ramène ainsi à un calcul d'intersection entre (hyper-)volumes que l'on traite de façon géométrique (détermination de l'intersection de volumes 4D). Cependant, dans le cas de formes d'objets simples, on peut parfois calculer analytiquement la collision : on cherche alors à écrire une équation où la seule inconnue est le temps et où les solutions sont les instants où se produit une collision. Cette technique est strictement équivalente à l'intersection 4D, mais elle est observée d'un point de vue non plus géométrique, mais analytique.

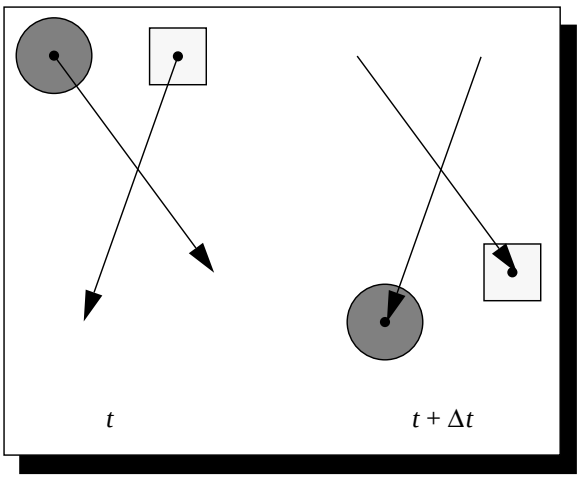

Figure 2. Exemple de collision non détectée

Dans cet article, nous faisons une distinction nette entre les algorithmes de détection (fine) des collisions et les algorithmes d'accélération de la détection. Les premiers ont pour but de fournir en sortie la position et l'instant exacts où la collision est survenue. Les seconds au contraire ont pour but d'éliminer les couples d'objets ou parties d'objets qui n'ont aucune chance d'entrer en collision et consiste alors en une

détection de non collisions certaines[1]. L'utilisation des deux phases est alors claire : on cherche d'abord à éliminer dans la phase d'accélération, un maximum de couple d'objets ou parties d'objets ne pouvant pas entrer en collision, afin de n'effectuer la détection fine, étape très coûteuse en calcul, que lorsqu'une collision est probable. En toute rigueur, la détection de collision comporte également une phase de traitement de la collision, que nous n'abordons pas dans cet article.

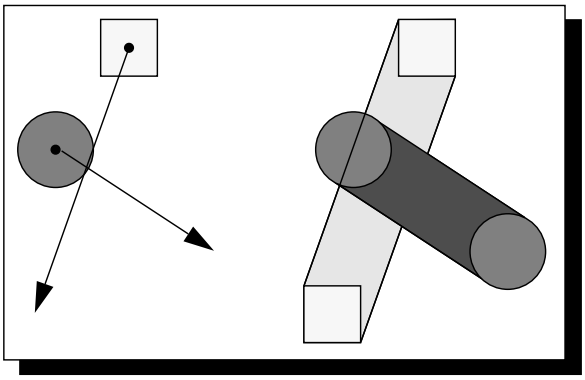

Figure 3. Intersection de volumes de trajectoire

3. Etat de l'art

3.1. Détection fine de la collision

Les techniques de calcul d'intersection qui entrent en jeu dépendent de la modélisation des objets. Ainsi, une détection pour des modèles polyédriques se fait en testant si une arête de l'un des objets passe par une facette de l'autre objet [Boy79]. De façon équivalente, ce test revient à examiner si un sommet de l'un des objets passe par une face de l'autre ou si la collision survient entre deux arêtes. Ainsi, une méthode basée sur une version 3D de l'algorithme de Cyrus-Beck et valable pour des polyèdres convexes a été proposée [MW88] : pour chaque sommet de l'un des objets, on teste s'il est à l'intérieur ou non de l'autre objet en lançant une demi-droite à l'infini et en comptant le nombre d'intersections avec les faces de l'autre objet. Un autre algorithme s'appuyant sur les quaternions a été implanté par [Can86]. Dans le cas des polyèdres, il est cependant possible de faire un calcul analytique [Boy79] où l'on considère séparément les trajectoires en translation et en rotation, toutefois la rotation rend les calculs très lourds. Dans le cas particulier des objets à faces triangulaires et où on ne s'intéresse qu'aux collisions entre sommets et faces, une méthode a été proposée par [MW88] et donne une équation du premier degré lorsque la facette est immobile et du cinquième degré lorsqu'elle est mobile et déformable.

Si par contre, les objets sont modélisés par des surfaces implicites, il faut échantillonner l'une des surfaces et pour chaque point obtenu, examiner s'il est à l'intérieur ou non de l'autre objet, par application de la fonction dedans-dehors associée à la surface. Ainsi plusieurs auteurs ont préconisé l'emploi de fonctions implicites [SP91][Gas93]. Il faut cependant remarquer qu'une collision fine nécessite un échantillonnage dense de la première surface. En outre, l'évaluation de la fonction dedans-dehors peut être coûteuse.

3.2. Accélération de la collision

L'accélération de la détection de collisions a été un sujet largement étudié, à la fois pour l'animation et les mondes virtuels. Afin d'obtenir une vue d'ensemble des divers algorithmes existants, nous proposons une classification des solutions par stratégies d'accélération : les quatre types dégagés sont les stratégies basées sur des subdivisions de l'espace, sur des boîtes englobantes, sur la topologie de la scène et enfin sur la vitesse des objets.

Une première catégorie de solutions consiste à subdiviser l'espace en zones. Deux objets situés dans des zones différentes ne peuvent pas s'interpénétrer. On y trouve les techniques de division régulière de

1. Certaines versions hiérarchiques des algorithmes d'accélération, qui se basent sur des subdivisions récursives de l'objet, peuvent déterminer quelles primitives, ou parties élémentaires des objets entrent en collision, mais cette phase devra être complétée par une calcul d'intersection exact.

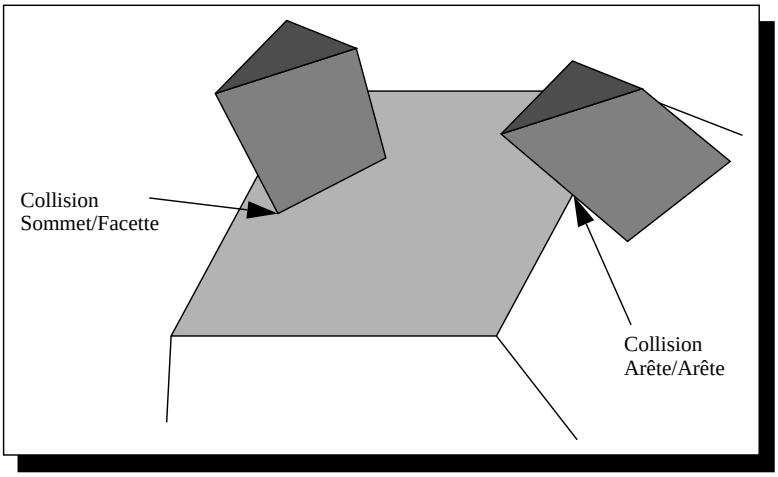

Figure 4. Collision Sommet/Facette et Arête/Arête

l'espace, mais aussi des algorithmes plus sophistiqués, à base de voxels [BT95] ou encore s'appuyant sur des tris de profondeur [FS91]. Dans la seconde catégorie nous mettons l'ensemble des méthodes à base de volumes englobants. La plupart du temps, on teste entre eux les volumes englobants de deux objets. Seule la méthode de [MW88] est différente puisqu'ils proposent une structure hiérarchique représentant la géométrie de l'objet afin de détecter si l'un des sommets de l'objet polyédrique appartient à la boîte englobante de l'autre objet. Dans tous les cas, le choix des volumes englobants employés est délicat : [Hub96] étudie par exemple une hiérarchie de sphères englobantes. [GSF94] emploient quant à eux, des boîtes englobantes. L'efficacité des comparaisons entre volumes englobants est mesurée avec la distance de Hausdorff qui mesure l'écart entre l'objet et le volume englobant. Le choix du volume est alors délicat : l'intersection de sphères englobantes est simple à tester mais ces volumes n'englobent pas l'objet de façon efficace. Si les boîtes englobantes ont le plus souvent une bonne distance de Hausdorff, l'intersection entre boîtes englobantes est très coûteuse en général. Seul le cas où les boîtes sont définies dans un même repère est simple à tester. Cette remarque conduit [GSF94] à faire les comparaisons dans divers repères, en calculant des boîtes englobantes de boîtes englobantes. Une meilleure solution a été avancée par [GLM96] qui proposent un algorithme efficace pour comparer deux boîtes englobantes quelconques.

Les deux catégories précédentes (zones et boîtes englobantes) partagent de nombreux points communs avec les techniques d'accélération du lancer de rayons. Les catégories suivantes sont plus spécifiques à l'accélération de la collision, car elles intègrent le facteur temps et plus particulièrement les évolutions des positions au cours du temps.

La troisième catégorie exploite des considérations topologiques. Dans ces méthodes, on évalue les distances qui séparent les objets entre eux. Si une distance devient négative, c'est que les objets se sont interpénétrés. Ainsi [CC86] proposent une méthode permettant d'évaluer la distance alors que [LC91] préfèrent un calcul incrémental de la distance entre objets polygonaux convexes obtenue à partir de la distance à l'instant précédent. Enfin, la quatrième catégorie englobe l'ensemble des stratégies basées sur la cinématique. On examine la vitesse relative entre deux objets : [Van94] propose de supprimer les objets qui s'éloignent les uns des autres tandis que [CK86] prédit l'instant où la collision peut avoir lieu à l'aide d'une borne inférieure de la distance et une borne supérieure de la vitesse relative entre deux objets. Cette méthode est enrichie par [LC92] et [DZ93] : une fois l'instant de la collision prédit, on stocke dans une pile les deux objets intervenant dans la collision. Cette pile est alors triée en fonction du temps : au sommet, on y trouve le couple d'objets qui entreront les premiers en collision. La simulation peut alors s'effectuer sans test de collisions jusqu'à cet instant.

4. Détection de collisions

Dans la suite de cet article, nous faisons l'hypothèse que les objets sont modélisés sous forme polyédrique. Cette hypothèse est justifiée car la plupart des modèles actuellement employés en animation, simulation et environnements virtuels sont polyédriques. Nous supposons de plus qu'entre deux instants de simulation, tous les points ont une trajectoire rectiligne uniforme.

4.1. Exposé des problèmes

La plupart des techniques de détection exposées dans l'état de l'art ci-dessus ne sont valables que pour des corps à géométrie fixe, c'est à dire pour des corps rigides. Or, les objets rigides ne sont qu'un cas particulier de l'ensemble des corps physiques.

Par ailleurs, dans le cas général d'une collision entre objets polyédriques, il ne suffit pas de calculer si un sommet de l'un passe par une face de l'autre. Il faut également vérifier si deux arêtes entrent en collision. La *figure 5* illustre ce problème en considérant (en vue de coupe) le cas particulier d'un objet en chute libre tombant sur un plan où la collision entre point et surface seule ne suffit pas. A l'instant *(a)*, le cube tombe en chute libre. A l'instant *(b)* on détecte une collision entre l'un des sommets du cube et le plan incliné. Le cube bascule et atteint la position décrite en *(c)*. S'il y a détection de collision entre les arêtes du cube et la bordure du plan, le cube reste en équilibre à sa position ou bascule autour du bord du plan. Sans détection de collisions entre arêtes, le cube continue son mouvement et se retrouve dans la position (d), immobile, car le point initial de la collision est considéré comme étant au dessus du plan et en collision avec lui.

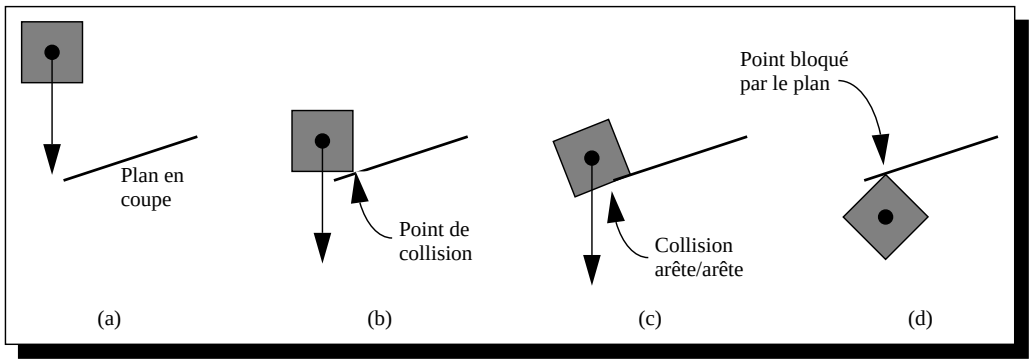

Figure 5. Cas de collision arête/arête critique (vue en coupe)

La méthode que nous proposons dans le paragraphe suivant permet à la fois la prise en compte des corps déformables et la détection des collisions entre arêtes.

4.2. Solution proposée

Afin de présenter notre algorithme, nous allons nous placer dans le cas d'une collision entre un objet rigide et un objet déformable. La collision entre deux objets rigides apparaît alors comme un cas particulier de cette collision et est donc compatible avec l'algorithme. Les collisions se réduisent alors aux cas suivants :

- *Sommet du corps déformable contre face du corps solide ;*
- *Arête du corps déformable contre arête du corps solide ;*
- *Sommet du corps solide contre face du corps déformable ;*

Nous allons étudier ces trois cas successivement, par ordre de complexité calculatoire. A chaque fois on considère les trajectoires des éléments (sommets, faces ou arêtes) et on détermine s'il existe un instant où les objets sont à la même position de l'espace.

- *Collision sommet du corps déformable / face du corps rigide*

En se plaçant dans le repère de la facette rigide, on peut ramener le calcul à celui d'un point passant par un plan fixe. En effet, si $\vec{\omega}$ est la vitesse de rotation instantanée du solide et $\vec{V}_G$ la vitesse instantanée du centre d'inertie alors on pose que la vitesse $\vec{V}_A$ au point A du corps déformable devient : $\vec{V'}_A = \vec{V}_A - \vec{V}_G + \overrightarrow{GA} \wedge \vec{\omega}$ (ce qui revient à étudier la vitesse relative du point A par rapport à la facette, en faisant une approximation de la rotation). Ensuite, on cherche s'il existe un instant t compris entre 0 et Δt, tel que : $A + t\vec{V'}_A \in$ facette. Ce test s'effectue en injectant l'expression $A + t\vec{V'}_A$ dans l'équation algébrique de la facette, qui est une équation du premier degré. On obtient alors une équation linéaire en t. La valeur de t obtenue est égale à la distance minimale du point A au plan rigide divisée par la composante normale au plan de la

vitesse relative $\vec{V}_A$. Il est intéressant d'utiliser dans une phase préliminaire l'orientation de la composante normale de la vitesse relative afin de déterminer si le point s'éloigne ou se rapproche de la facette. Dans le cas où il s'éloigne, il est inutile de calculer une intersection. Si la vitesse est bien orientée, on calcule la solution puis on vérifie qu'elle est bien entre 0 et Δt, on teste ensuite si le point d'intersection obtenu se trouve effectivement à l'intérieur de la facette (en effectuant une série de produits mixtes).

• *Collision arête du corps déformable / arête du corps rigide*

De la même façon que le calcul ci-dessus, on se place dans le repère du côté rigide *[CD]* et on transforme les vitesses des extrémités A et B du côté déformable comme exposé dans le premier cas, afin d'obtenir les vitesses relatives $\vec{V}_A$ et $\vec{V}_B$. On trouve alors qu'il y a collision si à un moment donné un point de l'arête déformable se trouve sur l'arête rigide. Mathématiquement, on cherche t entre 0 et Δt, u et w entre 0 et 1 tels que $A + t\vec{V}_A + u\left(B + t\vec{V}_B - A - t\vec{V}_A\right) = C + w\overrightarrow{CD}$. Cette expression se sépare en trois équations respectivement en x, y et z. On emploie l'une des équations afin de supprimer l'inconnue w, ce qui nous donne un système de la forme :

$$\begin{cases} a_0 t + a_1 u + a_2 tu + a_3 = 0 \\ a_4 t + a_5 u + a_6 tu + a_7 = 0 \end{cases}$$

u s'exprime alors comme une fonction homographique en t :

$$u = \frac{\alpha t + \beta}{\gamma t + \delta}$$

Une fois u éliminé, on obtient une équation de degré 2 en t. Il reste ensuite à vérifier que les valeurs de t, puis u puis w se trouvent dans leur bon domaine de définition.

• *Collision sommet du corps rigide / face du corps déformable*

Ce cas a déjà fait l'objet d'études. En effet, le problème est identique à celui développé par [MW88] pour le calcul de l'intersection entre deux corps déformables. La méthode proposée alors aboutissait à la résolution d'une équation de degré 5 nécessitant un algorithme itératif. Nous préférons traiter le problème de la manière suivante : si D est le point du corps rigide et A, B et C les sommets de la facette déformable, il y a intersection si et seulement si à un instant donné le point A appartient à la facette. Plutôt que d'exprimer l'équation du plan de la facette de façon paramétrique, on préfère l'exprimer sous forme algébrique. En d'autres termes, on cherche t compris entre 0 et Δt tel que : $\left(\overrightarrow{AB}(t) \wedge \overrightarrow{AC}(t)\right) \cdot \overrightarrow{AD}(t) = 0$, avec $\overrightarrow{AX}(t) = X + t\vec{V}_X - A - t\vec{V}_A$ pour $X = B$, C et D. Cette équation est du troisième degré en t. On cherche alors le plus petit t compris entre 0 et Δt, tel que le point d'intersection soit effectivement contenu par la face.

Dans les trois cas, on obtient un polynôme dont on peut trouver les racines en temps constant. La détection est alors immédiate : on teste chaque sommet du corps déformable avec chaque face du corps rigide, puis chaque arête de l'un avec chaque arête de l'autre et enfin chaque sommet du corps solide contre chaque face du corps déformable. La complexité de l'ensemble de la détection est donc quadratique.

Il nous reste à étudier le cas de la collision entre deux corps déformables. Il est possible d'exploiter les résultats précédents : en effet si on ne s'intéresse qu'aux collisions entre sommet et face déformable, le problème s'exprime de façon rigoureusement identique au dernier des trois cas présentés ci-dessus. Par contre, il semble que les collisions entre arêtes déformables ne peuvent pas être traitées en temps constant : si cela n'est pas crucial en animation précalculée, cela le devient pour des applications temps-réel. Cependant, ce type de collisions n'est peut-être pas aussi indispensable dans le cas de corps déformable (pas d'arêtes vives).

Ces routines forment la base de l'unité de traitement des collisions d'un système simulant la dynamique de corps déformables. Le moteur est destiné à reproduire les interactions mécaniques entre un organe et un outil rigide manipulé par le chirurgien, dans un simulateur pédagogique d'actes chirurgicaux. Le principe de la simulation est détaillé dans [MC97]. Les mesures ont été effectuées sur des modèles de corps déformables à nombre de nœuds croissants. La comparaison entre les mesures avec prise en compte des interactions (*figure*

6) ou non (*figure 7*) montrent que la détection des collisions représente 90% du temps de calcul de la simulation.

	R4600, 133MHz	P Pro, 200 Mhz	R10000, 194MHz
12 nœuds	630Hz	880Hz	2500Hz
120 nœuds	36Hz	60Hz	180Hz
258 nœuds	18Hz	25Hz	60Hz

Figure 6. Fréquence de simulation en fonction de la complexité du modèle

	R4600, 133MHz	P Pro, 200 Mhz	R10000, 194MHz
12 nœuds	3500Hz/82%	10680Hz/91%	14000Hz/82%
120 nœuds	330Hz/89%	1300Hz/95%	1450Hz/87%
258 nœuds	145Hz/87%	610Hz/95%	650Hz/90%

Figure 7. Fréquence de simulation sans détection de collision

5. Accélération de la détection

Les mesures présentées au paragraphe précédent montrent qu'une routine de détection des collisions doit être obligatoirement accompagnée d'une phase d'accélération pour être efficace. Notre choix se porte sur les méthodes utilisant les volumes englobants, car elles peuvent être hiérarchiques, ce qui permet de trouver le plus vite possible les parties d'objet à tester finement. Cependant, parmi les méthodes basées sur les boîtes englobantes exposées dans l'état de l'art, aucune ne tient compte du paramètre temporel et on obtient alors le problème évoqué précédemment sur la *figure 2*. En plus, ces méthodes d'accélération ne sont pas compatibles avec les objets déformables.

On s'intéresse donc à un procédé permettant de détecter rapidement si deux objets, rigides ou déformables sont susceptibles d'entrer en collision. Notre approche est la suivante : on suppose qu'à l'instant t_0 et à l'instant $t_0 + \Delta t$, on calcule des boîtes englobantes alignées sur les axes pour tous les corps. Pour les corps déformables, cette étape se fait en parcourant chaque nœud et en déterminant quelles sont les plus petites et les plus grandes valeurs pour chaque coordonnée x, y et z. Pour les corps rigides, on peut simplement considérer une boîte englobant la boîte englobante du corps dans son repère local. Comme les trajectoires entre deux instants sont supposées rectilignes uniformes, on considère le tronçon de pyramide (en 4D) dont les deux bases sont les deux boîtes englobant l'objet aux deux instants. La *figure 8* montre, pour une seule dimension comment on détecte la collision probable de deux objets.

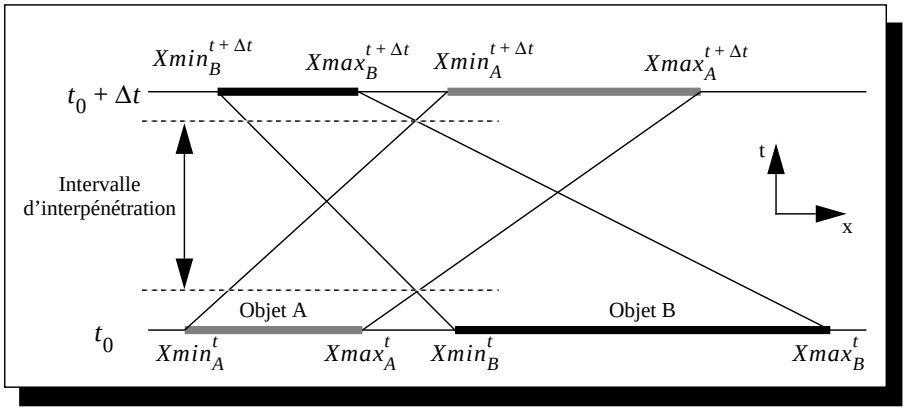

Figure 8. Intersection de trajectoires de boîtes englobantes

On retrouve le même type de schéma pour y et z. Afin d'alléger les notations, et sans perte de généralité, on pose $t_0 = 0$. De plus, on note $\Delta Xmin_A = Xmin_A^{\Delta t} - Xmin_A^0$, de même pour $\Delta Xmax_A$, et pour l'objet B. A un instant t, la boîte de A est définie par la borne inférieure suivante :

$$Xmin_A^t = Xmin_A^0 + t \times \Delta Xmin_A$$

De façon analogue on obtient sa borne supérieure. A l'instant t, les deux boîtes ne s'intersectent pas si l'une des deux conditions suivantes est vérifiée :

$$Xmin_A^t > Xmax_B^t \text{ ou } Xmin_B^t > Xmax_A^t \qquad (1)$$

En exprimant ses termes en fonction de t, et en effectuant les mêmes calculs pour les trois dimensions, on obtient six expressions linéaires en t qui restent positives tant qu'il n'y a pas d'intersection. Par exemple :

$$(\Delta Xmin_A - \Delta Xmax_B)\, t + \left(Xmin_A^0 - Xmax_B^0 \right) \geq 0 \qquad (2)$$

La mise en œuvre de l'algorithme est simple. La résolution de chaque inégalité donne un instant correspondant soit à l'entrée en collision, soit à la séparation des deux corps. Par exemple, dans l'équation *(2)*, on trouve que :

$$t_{sol} = \frac{Xmin_A^0 - Xmax_B^0}{\Delta Xmin_A - \Delta Xmax_B}$$

Les deux cas se distinguent par le signe de $\Delta Xmin_A - \Delta Xmax_B$. Si l'expression est positive, cela signifie que la condition de non collision *(1)* est valable pour $t \geq t_{sol}$ ce qui signifie que la collision prend fin à l'instant calculé, sinon, elle débute à cet instant là.

On obtient donc suivant les cas, une borne supérieure ou inférieure de l'intervalle de temps où se produit l'interpénétration. On parcourt ensuite les inégalités, chaque inégalité une fois résolue modifie la borne supérieure ou inférieure de l'intervalle d'interpénétration. Sitôt que l'intervalle devient vide où qu'il n'est plus compris entre 0 et Δt , on conclut qu'il n'y a pas d'intersection et l'algorithme est terminé. En moyenne, il suffit donc de parcourir trois inégalités sur les six.

La *figure 9* montre que notre proposition d'accélération apporte un gain moyen de 20% par rapport à la *figure 7* lorsque les objets entrant en collision sont suffisamment complexes.

	R4600, 133MHz	P Pro, 200 Mhz	R10000, 194MHz
12 nœuds	640Hz	900Hz	2600Hz
120 nœuds	40Hz	61Hz	180Hz
258 nœuds	24Hz	30Hz	67Hz

Figure 9. Fréquence de simulation avec la routine d'accélération des collisions

6. Bilan - Perspectives

Les algorithmes de détection et d'accélération proposés ont été testés dans un moteur de simulation basée sur la mécanique. Cependant, ces routines sont génériques et peuvent être utilisées en animation en général ainsi que dans les mondes virtuels. Pour la détection de collision nous proposons une méthode originale pour prendre en compte les collisions entre arêtes. Dans le cadre de l'accélération, la solution décrite se base sur les trajectoires lors des tests entre boîtes englobantes. Les diverses méthodes de détection et d'accélération présentées sont compatibles avec les corps déformables.

Cependant, il reste des limites. Tous les cas de collisions entre corps rigides ou déformables sont calculés, sauf la collision entre arêtes déformables qui exige un algorithme itératif, qui est coûteux dans le cas d'animation calculée et strictement à prohiber dans le cas temps-réel. L'algorithme d'accélération décrit est un bon moyen de détection intermédiaire, afin de déterminer rapidement si deux objets sont susceptibles de s'interpénétrer ou non. Cependant, il ne suffit pas car, si une collision possible est détectée, il va falloir tester toutes les primitives géométriques des objets entre eux. Notre accélération doit être complétée par une

méthode dont le but est d'affiner les recherches afin de déterminer les primitives de base en collision. La méthode proposée peut être hiérarchique, mais cette hiérarchie est coûteuse dans le cas des corps déformables. Il s'agit de trouver un compromis entre la gestion de la hiérarchie et la précision de l'algorithme. Des études en cours montrent que l'on obtient une accélération remarquable en employant l'algorithme de [GLM96] sur des corps rigides. Il serait très intéressant de l'adapter aux corps déformables.

7. Références

[Boy79] Boyse, J. W., "Interference Detection among Solids and Surfaces" *Communications ACM*, vol 22, 1, janvier 1979, pp 3-9.

[BT95] Bandi, S., et Thalmann, D., "An Adaptative Spacial Subdivision of the Object Space for Fast Collision Detection of Animated Rigid Bodies" *Proceedings of the Eurographics'95 Conference, Computer Graphics Forum*, 14, 3, Maastricht, 28 août-1 septembre 1995, pp 259-270.

[Can86] Canny, J., "Collision Detection for Moving Polyhedra" *IEEE Transactions on Pattern Analysis and Machine Intelligence*, mars 1986, vol 8, no 2, pp 200-209.

[CC86] Cameron, S.A., et Culley, R.K., "Determining the Minimum Translational Distance between Two Convex Polyhedra" *Proceedings of the IEEE International Conference on Robotics and Automatons*, 1986, pp 591-597.

[CK86] Culley, R.K., et Kempf, K.G., "A Collision Detection Algorithm Based on Velocity and Distance Bounds" *Proceedings of the IEEE International Conference on Robotics and Automatons*, 1986, pp 1064-1069.

[DZ93] Dworkin, P., et Zelter, D., "A New Model for Efficient Dynamic Simualtion" *Eurographics Workshop on Animation and Simulation*, Barcelone, 4-5 septembre 1993, pp 135-147.

[FS91] Forgue, M.C., et Shinya, M., "Real-Time Impact Dynamics Simulation" *Compugraphics* 1991, pp 427-435.

[Gas93] Gascuel, M.P., "An Implicit Formulation for Precise Contact Modeling between Flexible Solids" *SIGGRAPH'93 Conference Proceedings, Computer Graphics annual conference series*, Anaheim, 1-6 août 1993, pp 313-320.

[GLM96] 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.

[GSF94] Garcia-Alonso, A., Serrano, N., et Flaquer, J., "Solving the Detection Problem" *IEEE Computer Graphics and Applications*, Mai 1994, pp 36-43.

[Hub96] Hubbard, P.M., "Approximating Polyhedra with Spheres for Time-Critical Collision Detection" *ACM Transactions on Graphics*, 15, 3, juillet 1996, pp 179-209.

[LC91] Lin, M.C., et Canny, J.F., "A Fast Algorithm for Incremental Distance Calculation" *Proceedings of the IEEE International Conference on Robotics and Automatons*, avril 1991, Sacramento, California, pp 1008-1015.

[LC92] Lin, M.C., et Canny, J.F., "Efficient Collision Detection for Animation" *Eurographics Workshop on Animation and Simulation*, Cambridge, 7-11 septembre 1992.

[MC97] Meseure, P., et Chaillou, C., "Deformable Body Simulation with adaptive subdivision and cuttings" *WSCG'97*, Plzen, 10-14 février 1997 (acceptée).

[MW88] Moore, M., et Wilhelms, J., "Collision Detection and Response for Computer Animation" *SIGGRAPH'88 Conference Proceedings, Computer Graphics*, 22, 4, Atlanta, 1-5 Août 1988, pp 289-298.

[SP91] Sclaroff, S., et Pentland, A., “Generalized Implicit Functions For Computer Graphics” *SIGGRAPH’91 Conference Proceedings, Computer Graphics*, 25, 4, Las Vegas, 28 juillet-2 août 1991, pp 247-250.

[Van94] Vanecek, G., “Back Face Culling Applied to Collision Detection of Polyhedra” *The Journal of Visualization and Computer Animation*, vol 5, 1994.

[Vrml96] Silicon Graphics Inc., The Virtual Reality Modeling Langage Specification, version 2, *http://vrml.sgi.com/moving-worlds/spec/index.html*, Août 1996.