Un Algoritmo Eficiente para la Detección de Intersecciones en Tiempo Real

Olmedo Arcila, Carlos Muñoz y José María Bañón
Escuela de Ingeniería de Sistemas y Computación
Universidad del Valle, Cali, Colombia.
E-mail : { olarcila, carmunoz, banon}@eisc.univalle.edu.co

Abstract

This paper presents a new solution to a practical and efficient interference detection algorithm between polygonal models. Most efficient interference detection algorithms use hierarchical bounding volumes to reduce the number of calls to expensive collision tests between polygons. We propose an hierarchical scheme based on sphere trees that involves outer and inner spheres. The algorithm takes advantage on the important role of inner spheres for detecting the existence of intersections. Several heuristics have been proposed to improve the performance of the interference detection algorithm. Some experiments have been carried out that demostrate the real time performance of the algorithm.

Keywords: Computer graphics, collision detection, hierarchical bounding volumes, sphere trees.

Resumen

Este artículo presenta una nueva solución para un algoritmo práctico y eficiente de detección de intersecciones entre modelos poligonales. Los algoritmos más eficientes de detección de intersecciones utilizan volúmenes limitantes jerárquicos para reducir el número de llamadas a costosos tests de colisiones entre polígonos. Proponemos un esquema jerárquico basado en árboles de esferas que considera esferas externas e internas. El algoritmo utiliza el importante papel de las esferas internas para detectar la existencia de intersecciones. Varias heurísticas han sido propuestas para mejorar el desempeño de del algoritmo de detección de intersecciones. Algunos experimentos han sido realizados que demuestran el desempeño en tiempo real del algoritmo.

Palabras Clave: Computación gráfica, detección de colisiones, volúmenes limitantes jerárquicos, árboles de esferas.

1 Introducción

La detección de colisiones en tiempo real para objetos poliédricos es un problema fundamental en muchas aplicaciones gráficas interactivas en las que se requiere verificar un número grande de collisiones por segundo. Por ejemplo, en cualquier simulación dinámica entre objetos rígidos en realidad virtual es necesario verificar constantemente si los objetos interseccionan entre ellos [10]. En los algoritmos prácticos de planificación del movimiento de robots, la detección de colisiones debe proporcionar una respuesta en tiempo real pues es una componente crucial para lograr que el robot siga una trayectoria segura sin colisión con los objetos del entorno [12]. En animación, la detección de colisiones permite que el movimiento de los objetos sea más natural evitándose la superposición de objetos. En general, en las aplicaciones prácticas señaladas, la detección de colisiones es la componente que más tiempo consume.

Evidentemente, para resolver el problema de la detección de colisiones que considera que los objetos están en movimiento, ha de ser resuelto primero el problema particular de la detección de intersecciones en donde los objetos se consideran en reposo. El principal reto de un detector de intersecciones práctico es diseñar un algoritmo con la capacidad de decidir rápidamente la intersección entre dos objetos con geometría compleja. Muchos modelos de detección de intersecciones han sido propuestos en la literatura [13, 5, 9, 19, 1, 14, 4, 11, 17]. La solución más exitosa para acelerar el desempeño de los detectores de intersecciones ha sido la de envolver los objetos en jerarquías de volúmenes limitantes consistentes en primitivas geométricas sencillas que conducen a eficientes tests de solapamiento. Los tests de solapamiento son útiles para garantizar la ausencia de intersección y limitan el número de ejecuciones del cálculo exacto de la intersección entre los cuerpos.

El objetivo principal del artículo es el desarrollo de un algoritmo rápido de detección de intersecciones válido para ser utilizado en aplicaciones de tiempo real y entre objetos arbitrarios de geometría compleja. Se propone un detector de intersecciones que utiliza un árbol binario de esferas exteriores e interiores del objeto propuesta por Muñoz et al [15], el cual ha demostrado mejoras en su desempeño respecto a trabajos anteriores. La principal ventaja de la utilización de esferas interiores es que permiten la detección de intersecciones evitándose llamadas inútiles a tests exactos de intersección. Se presentan resultados experimentales que muestran el desempeño del detector de intersecciones. El artículo está organizado de la manera siguiente. En la sección 2 se resume los trabajos previos más significativos. La sección 3 describe la estructura jerárquica de volúmenes limitantes utilizada en este trabajo. En la sección 4 se da una descripción general de los algoritmos de detección de intersecciones que utilizan una estructura jerárquica de volúmenes limitantes y se propone una manera de mejorarlos. En la sección 5 se describe el algoritmo propuesto en este trabajo. En la sección 6 se presenta una análisis cuantitativo del desempeño en tiempo real del algoritmo. Finalmente, en la sección 7 se presentan las principales conclusiones.

2 Trabajos Previos

El problema de la detección de intersecciones ha sido objeto de numerosas publicaciones en la literatura. Para objetos poliédricos generales los algoritmos de detección de intersecciones más eficientes se basan en la utilización de estructuras jerárquicas de volúmenes limitantes las cuales permiten reducir el número de tests de intersecciones. Tales algoritmos se diferencian unos de otros por la primitiva geométrica utilizada como volumen limitante. Por su simplicidad y puesto que conducen a eficientes tests de solapamiento las esferas y las cajas son las primitivas más frecuentemente utilizadas en los detectores de intersecciones.

Algunos autores [13, 5, 19, 17, 8] desarrollan detectores de intersecciones basados en árboles octrees de esferas. El árbol octree es una representación de objetos obtenida mediante una subdivisión recursiva del espacio ocupado por el objeto en octantes. El árbol octree de esferas se construye a partir de las esferas circunscritas a los octantes del árbol octree ocupados por el objeto. Tales árboles permiten desarrollar detectores de intersecciones que son rápidos y fáciles de implementar. El inconveniente de estos modelos es que no producen jerarquías de esferas que aproximen estrechamente al objeto.

Un esfuerzo considerable ha sido realizado en el desarrollo de algoritmos de detección de intersecciones de manera que se cumplan los siguientes criterios: i) que los volúmenes limitantes se adapten estrechamente a la forma del objeto, ii) que el test de solapamiento entre los volúmenes limitantes sea rápido, y iii) que el coste de actualización de los volúmenes limitantes en caso de rotación ó traslación del objeto sea rápido. Para ello, numerosas heurísticas para obtener volúmenes limitantes que satisfagan los anteriores criterios han sido desarrolladas basadas en métodos de particionamiento adaptativo de la forma del objeto. A continuación se señalan los trabajos más significativos.

Árboles jerárquicos de esferas, basados en técnicas adaptativas, han sido propuestos por varios autores [9, 14, 15, 18]. En general tales árboles se adaptan mejor al objeto que los árboles octrees de esferas. La eficiencia de tales árboles depende mucho del algoritmo particular de generación de esferas. Representan un compromiso entre la precisión de la

representación y el costo computacional. Held et al. [7] proponen jerarquías de volúmenes limitantes basadas en cajas alineadas con los ejes (AABB). Gottschalk et al. [4] presentan un algoritmo RAPID de detección de intersecciones basado en jerarquías de cajas orientadas (OBB). La idea principal de la estructura OBB es la utilización de cajas alineadas en una dirección tal que el volumen inutil de la caja sea mínimo. Klosowski et al. [11] desarrolla un algoritmo basado en politopos de orientaciones discretas (k-dop) los cuales consisten en poliedros convexos cuyas caras son determinadas por pares de caras paralelas cuyas normales exteriores tienen k orientaciones predeterminadas. La ventaja principal de los modelos OBB y k-dop es que aproximan estrechamente al objeto, pero con el inconveniente de que tanto los tests de solapamiento como las actualizaciones de sus nodos son costosas.

3 El Modelo de Arbol de Esferas

Muñoz et al. [15] desarrollan un modelo de estructura jerárquica de volúmenes limitantes basada en esferas el cual ha sido utilizado en este trabajo para el desarrollo del detector de intersecciones. A continuación se resume las características más relevantes del modelo. El modelo se aplica a objetos cuya geometría es la de un prisma recto con base un polígono de lados rectos y curvos. La jerarquía de esferas está organizada en un árbol binario cuya raiz es la esfera mínima que contiene al objeto. El árbol consiste en dos tipos de nodos: i) las esferas exteriores que cubren la superficie del objeto y ii) las esferas interiores que aproximan el interior del objeto. El árbol es generado mediante un proceso de subdivisión binaria recursiva del prisma. El proceso se realiza de forma que la parte del objeto cubierta por una esfera es también cubierta por los subárboles derecho e izquierdo de la esfera. Por ello, todas las esferas exteriores pertenecientes a un mismo nivel del árbol forman una representación exterior de la superficie del objeto. Cuanto más profundo es el nivel del árbol, la representación exterior correspondiente aproxima al objeto con una mejor resolución. Las esferas interiores son nodos terminales del árbol. Cuanto mayor es la altura del árbol las representaciones exteriores tienden hacia la superficie del objeto y las esferas interiores tienden hacia el interior del objeto con error cero. Muñoz et al. [15] presentan comparaciones del desempeño del modelo con otras representaciones previas las cuales son favorables al modelo. Las características más importantes del modelo son: i) es rápido, ii) se adapta estrechamente al objeto, y iii) contiene esferas interiores las cuales tienen un papel importante en la detección de colisiones. Se obtienen árboles binarios de 50.000 esferas totales en 15.34 seg. para una lámpara compuesta por 108 caras poligonales. En general, se ha criticado a los árboles de esferas el no adaptarse estrechamente a los objetos que representan y ello ha sido el origen del desarrollo de otros volúmenes limitantes. El modelo desarrollado por Muñoz et al. [15] obtiene representaciones esféricas en las que las esferas están en gran parte dentro del objeto y por tanto el volumen inutil de la esfera que queda por fuera del objeto es pequeño. Esta característica es muy deseable para su aplicación a un detector de intersecciones.

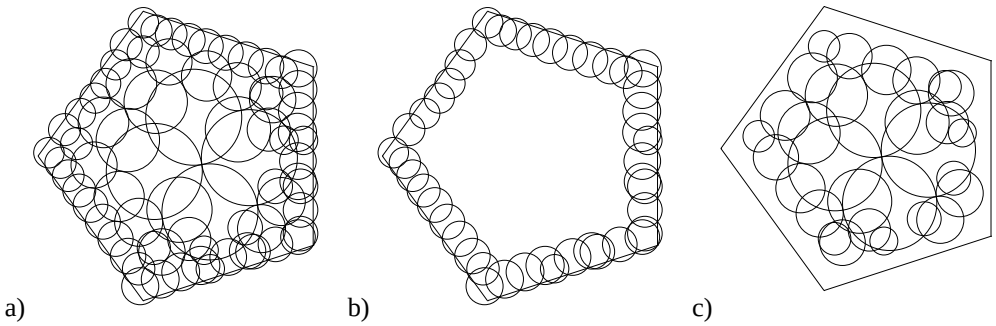


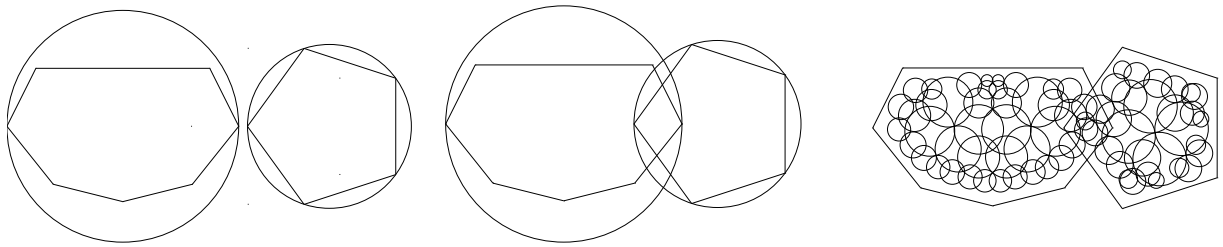
Fig. 1: Árbol binario de un polígono convexo.

El modelo de árbol de esferas consiste en esferas exteriores e interiores al objeto. Se trata pues de dos representaciones diferentes en el mismo árbol binario de esferas: i) una representación de superficie compuesta por esferas exteriores y ii) una representación de volumen compuesta por esferas interiores al objeto. Para facilitar la visualización del árbol binario, en la figura 1 se muestra un ejemplo bidimensional del modelo. Se trata del árbol binario de círculos de un polígono. En la figura 1.a) se muestra el polígono junto a todos los círculos que lo recubren; las figuras 1.b) y 1.c) muestran respectivamente la representación exterior e interior. Nótese cómo el volumen inutil de los círculos exteriores es una pequeña fracción de la superficie de los mismos. Normalmente las estructuras jerárquicas utilizadas en la detección de intersecciones son aproximaciones de superficie que son útiles para detectar la ausencia de detecciones pero no detectan las intersecciones.

La incorporación de esferas interiores permite la detección eficiente de intersecciones como se explica en la siguiente sección.

4 El Esquema Jerárquico en la Interseción de Colisiones

La principal ventaja de la utilización de volúmenes limitantes jerárquicos en la detección de intersecciones es la detección rápida de la ausencia de intersección entre dos objetos. Ello se realiza verificando si existe solapamiento entre cada par de nodos de los árboles que aproximan los objetos. Si dos nodos no se solapan entonces las partes cubiertas por dichos nodos tampoco interseccionan y no hace falta seguir verificando solapamiento entre los nodos hijos. Por el contrario, si dos nodos se solapan entonces es necesario seguir verificando si hay solapamiento entre los nodos hijos.



(a) (b) (c)

Fig. 2: Detección de Colisiones Entre dos Cuerpos.

Las ideas anteriores se ilustran en las figuras 2.a) y 2.b). En la figura 2.a) la ausencia de intersección entre los dos objetos aparece por el no solapamiento entre las representaciones exteriores de los mismos. En la figura 2.b) hay solapamiento entre las representaciones exteriores, lo cual indica que hay que seguir verificando por solapamientos entre niveles hijos. En general, pero no siempre, los algoritmos de detección de intersecciones construyen los árboles jerárquicos de volúmenes limitantes de forma que los nodos terminales representen a los polígonos que constituyen la superficie de los objetos. Cuando no se encuentra ausencia de solapamiento entre pares de nodos se llega finalmente a los nodos terminales en donde se verifica si hay ó no intersección entre los pares de polígonos que representan.

El esquema algorítmico anterior es común a muchos detectores de intersecciones y tiene un coste total T que viene dado por la fórmula siguiente:

$$T = N_v \cdot C_v + N_u \cdot C_u + N_p \cdot C_p$$

donde N_v es el número de tests de solapamientos entre pares de volúmenes limitantes, C_v es el coste del solapamiento de un par de volúmenes limitantes, N_u es el número de volúmenes limitantes que se actualizan, C_u es el coste de la actualización de un volumen limitante, N_p es el número de tests de intersección entre polígonos, C_p es el coste de la intersección entre dos polígonos. Evidentemente, los parámetros de coste dependen del volumen limitante utilizado. La ventaja de utilizar como volumen limitante la esfera es que C_u es cero y C_v es muy pequeño, sin embargo los números N_v y N_p suelen ser típicamente más grandes que en otros tipos de volúmenes limitantes.

El esquema anterior es ventajoso tanto si hay como si no hay intersección. En el primer caso, se reduce el número de tests de intersección entre pares de polígonos de los objetos. En el segundo caso, la ausencia de intersección se realiza rápidamente mediante algoritmos eficientes de solapamiento entre volúmenes limitantes. Este esquema general puede ser mejorado si se introducen volúmenes que aproximan al objeto por el interior. Supongamos que dos objetos están aproximados por volúmenes interiores entonces el solapamiento entre cualquier par de volúmenes interiores implica necesariamente una intersección entre los respectivos objetos. La idea principal es pues la de utilizar los volúmenes interiores para detectar la existencia de colisiones. Entonces, si se toman como volúmenes interiores primitivas geométricas sencillas, tales como los volúmenes limitantes que conducen a eficientes tests de solapamiento entre ellos, se gana rapidez en la detección de intersecciones. La figura 2.c) ilustra cómo el solapamiento entre las representaciones esféricas internas de dos objetos garantiza la intersección entre ellos. En la fórmula del costo anterior esto equivale a disminuir el número de intersecciones entre polígonos N_p a costa de aumentar el número de solapamientos N_v entre esferas las cuales tienen un costo $C_v << C_p$. Evidentemente, la eficiencia de la utilización de volúmenes internos en la detección de intersecciones dependerá esencialmente del volumen ocupado por dichos volúmenes en el interior del objeto. Cuanto mayor sea el

volumen interior del objeto ocupado por los volúmenes interiores mejor será el desempeño de tales volúmenes en la detección de intersecciones. Sin embargo, hay que tener también en cuenta que para que los volúmenes interiores aproximen eficientemente el interior del objeto es necesario aumentar la profundidad del árbol binario lo cual puede representar un costo computacional importante. En la práctica hay un compromiso entre la profundidad del árbol de esferas y la rapidez de la detección de intersecciones que debe resolverse ya sea por heurísticas ó teniendo en cuenta las características del caso particular.

5 El Algoritmo

Se ha desarrollado un algoritmo de detección de intersecciones utilizando el árbol binario de esferas propuesto por Muñoz et al. [15]. El algoritmo procede de la manera siguiente. En un primer preprocesamiento se calculan los árboles binarios de esferas de los dos objetos hasta la resolución requerida. Seguidamente se procede a recorrer por profundidad y simultáneamente los dos árboles para realizar los tests de solapamiento entre pares de nodos. En este caso, los nodos corresponden a esferas externas ó internas. Si dos esferas externas se solapan entonces se procede a recorrer simultáneamente el subárbol correspondiente a cada nodo junto con los subárboles hijos del otro nodo. Si las dos esferas externas no se solapan entonces no se procede a ningún recorrido con los subárboles hijos. Si una esfera externa de un objeto se solapa con una esfera interna del otro objeto se procede al recorrido simultáneo de los subárboles hijos de la esfera externa con la esfera interna. Si una esfera interna de un objeto se solapa con una esfera interna del otro objeto es obvio que se ha detectado una intersección entre ambos objetos. Entonces el algoritmo retorna interferencia entre los objetos.

Con el objeto de hacer más eficiente el algoritmo se ha introducido una heurística que consiste en dar prioridad a los tests de solapamiento entre esferas que estén próximas entre sí. Ello con el objeto de detectar intersección entre los objetos lo antes posible. Cuando las esferas de dos nodos se solapan y se procede al recorrido simultáneo del subarbol de uno de los nodos con los subárboles hijos del otro nodo se dá prioridad al recorrido con el subárbol hijo más cercano. Aquí, la distancia entre subárboles se entiende referida a la distancia entre los centros de las esferas de las raíces de los subárboles.

Las alturas de los árboles binarios de esferas que se necesitan para el detector de intersecciones depende esencialmente del tamaño relativo entre los objetos y de la distancia entre ellos. En el caso en el que los objetos estén alejados pocas esferas son necesarias. Si por el contrario los objetos están muy próximos entonces para detectar la ausencia de intersección se necesita que el diámetro de las esferas exteriores de las hojas de los árboles sean del mismo orden de magnitud que la distancia que separa los objetos. Puesto que la distancia entre ellos es pequeña entonces será necesaria una altura grande para que las esferas de las hojas tengan el diámetro adecuado. En el caso en que los objetos se penetren entonces las alturas dependen de la magnitud de la penetración. Si la penetración es muy pequeña entonces será necesario que la altura sea grande para que existan esferas interiores pequeñas que se solapen entre ellas en la zona común. Una estimación grosera de la proximidad entre dos objetos nos la proporciona la distancia de solapamiento $d_{sol} = |R_1 + R_2 - d|$ entre las esferas mínimas de los objetos donde R_1 y R_2 son los radios de las esferas y d es la distancia entre sus centros. En la práctica es importante disponer de heurísticas conducentes a determinar rápidamente la altura idónea del árbol binario en función de la situación particular. Una forma de determinar la altura de los árboles binarios es teniendo en cuenta que para que las esferas internas de los dos objetos se solapen en una región cuyo tamaño es del orden de d_{sol} debe haber en dicha región esferas internas de ambos objetos. Para ello, podemos imponer que d_{sol} sea un múltiplo de la resolución d_{res} del árbol, la cual viene medida por el diámetro promedio de las esferas exteriores (ver figura 3). Es decir que se debe descender en la generación del árbol hasta una altura en la que el diámetro promedio de las esferas exteriores sea del orden de $\frac{d_{sol}}{n}$, donde $n \geq 3$ es un entero. En la siguiente sección se da una estimación cuantitativa de esta estrategia.

6 Resultados Numéricos

El algoritmo de detección de intersecciones ha sido implementado en el lenguaje **C++** en una plataforma PC-Linux con Pentium XEON 1700 MHz. Para estudiar el desempeño del algoritmo desarrollado se han realizado varios experimentos. Los objetos utilizados en los experimentos junto con su representación esférica se muestran en las figura 4.

Un primer experimento ha sido realizado para medir el tiempo empleado por el algoritmo en detectar la intersección para varias combinaciones de pares de objetos. Varios autores [7, 20] han señalado la dificultad en la comparación entre los tiempos de ejecución de los algoritmos de detección de colisiones. En general, los tiempos de respuesta que se obtienen dependen muy sensiblemente del escenario particular en que se realiza el test de intersección. Así por ejemplo, dependiendo del tamaño relativo y de la posición relativa entre los objetos se obtienen tiempos de respuesta que pueden ser muy diferentes. Con el objeto de obtener un tiempo de intersección promedio que tenga en cuenta las posiciones relativas entre los objetos se han realizado promedios entre 100000 tests de intersecciones para cada par de objeto escogido. El

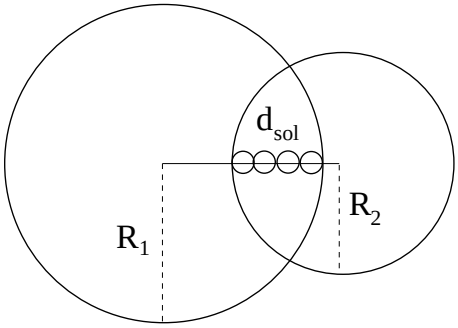


Fig. 3: Esferas en la región de solapamiento.

test de intersección se realiza en un volumen cúbico cuyo lado es aproximadamente cinco veces el tamaño de los objetos escogidos. Los objetos son colocados en el escenario en donde son sometidos a traslaciones y rotaciones aleatorias en el interior del volumen. En este experimento, los árboles binarios de los objetos han sido calculados hasta el nivel $n = 12$. En la tabla 1 se dan los resultados del tiempo promedio en la detección de intersección (**TI**) y del tiempo promedio en la detección de la ausencia de intersección (**TNI**) para varios pares de objetos. En el caso particular de la intersección esfera-esfera ambos tiempos son muy similares pues tanto la intersección como la ausencia de detección se determina trivialmente comparando la distancia entre los centos de las esferas con la suma de los radios. En general, como puede apreciarse en la tabla 1, para las colisiones en las que no interviene la esfera se obtienen tiempos de respuesta entre 25 y 78 microsegundos entre pares de objetos con un total de entre 16 y 204 vértices, lo cual corresponde a la detección de la intersección en tiempo real. Cuando se tiene en cuenta la intersección entre la esfera y cualquier otro de los objetos el tiempo de detección de intersección es menor, debido a que la esfera se representa por si misma de manera exacta y por tanto no hace falta aproximarla. De entre los pocos trabajos que utilizan representaciones esféricas en la detección de intersecciones y reportan resultados numéricos destacamos el de Tzafestas y Coiffet [19] quienes proponen un detector de intersecciones para aplicaciones en realidad virtual y en tiempo real. El detector de intersecciones está basado en el árbol octree de esferas del objeto y lo aplican a una escena virtual que consiste en varios objetos poliédricos entre 8 y 20 vértices. Obtienen tiempos de detección de intersecciones del orden de 20 milisegundos en una plataforma HP715 con 50 MHz.

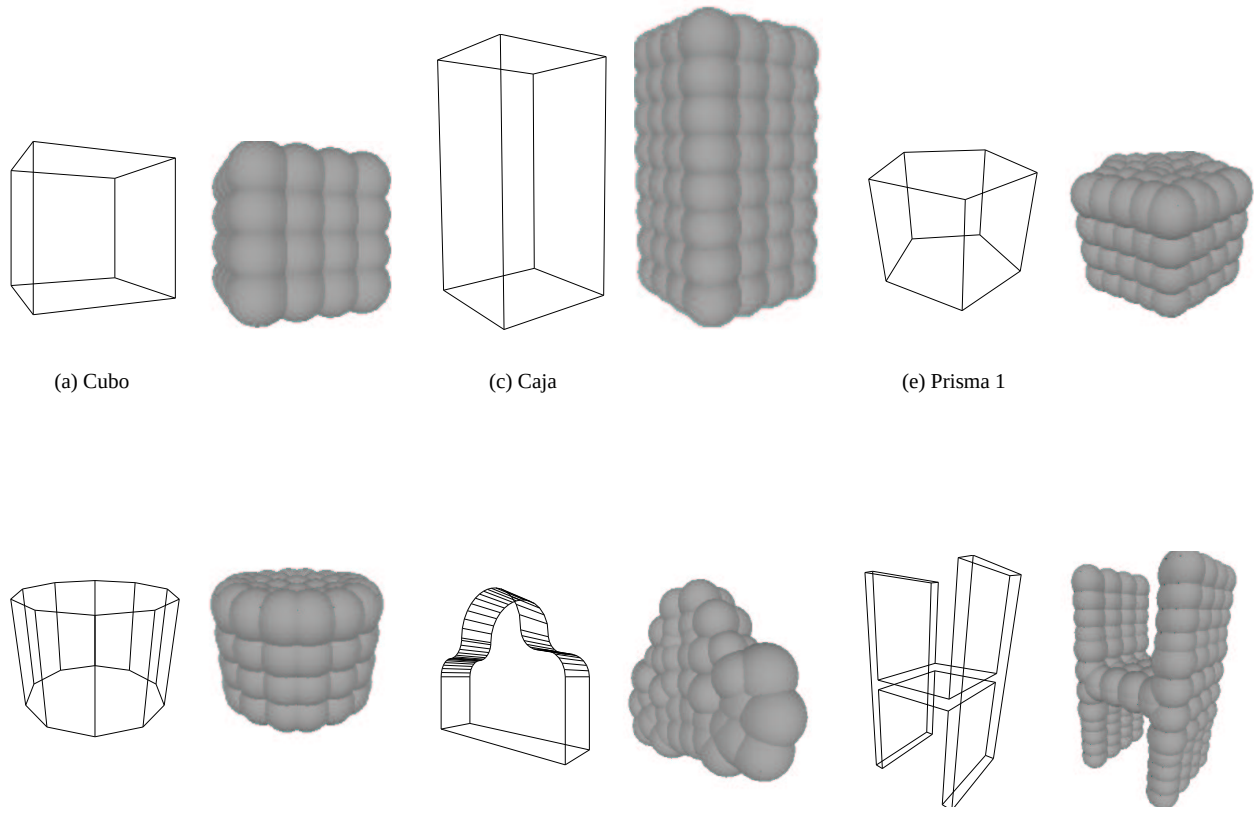
Objeto - Objeto	**T I**	**T N I**
Esfera-Esfera	0.75	0.61
Esfera-Cubo	5.70	0.60
Esfera-Caja	7.18	0.58
Esfera-Prisma 1	5.69	0.60
Cubo-Cubo	24.52	1.26
Cubo-Caja	23.86	1.85
Caja-Caja	29.03	1.70
Prisma 1-Prisma 1	25.69	0.96
Cubo-Prisma 1	21.98	0.84
Caja-Prisma 1	25.59	1.24
Prisma 1-Prisma 2	25.41	0.85

Objeto - Objeto	**T I**	**T N I**
Cubo-Prisma 2	23.77	1.53
Caja-Prisma 2	26.09	2.08
Prisma 2 -Prisma 2	27.4	1.84
Cubo-Puerta	38.59	0.57
Prisma 1-Puerta	36.5	0.91
Prisma 2-Puerta	48.68	0.63
Puerta-Puerta	52.01	0.85
Cubo-H	47.19	1.33
Prisma 1-H	45.7	1.53
Prisma 2-H	44.6	1.64
H-H	77.59	2.05

Tabla 1: Tiempos en la detección de la intersección.

Un segundo experimento se ha realizado con el objetivo de evaluar el efecto que tiene sobre el tiempo promedio la heurística que considera el camino más corto entre los subárboles en el recorrido simultáneo de los árboles binarios. Para ello se han realizado promedios del tiempo de intersección con y sin la heurística mencionada en el mismo escenario anterior. Los resultados se muestran en la tabla 2 en donde se indican los tiempos utilizados por el algoritmo sin la heurística (SH) y con la heurística (CH) para varios pares de objetos. El efecto de la heurística es importante pues permite reducir entre 4 y 5 veces el tiempo de cálculo de la detección de intersecciones.

Un tercer tipo de experimento ha sido realizado con el objetivo de analizar la heurística mencionada en la sección 5 para



(a) Cubo (c) Caja (e) Prisma 1

(g) Prisma 2 (i) Puerta (k) H

Fig. 4: Objetos Poliedricos.

generar automáticamente las alturas de los árboles de dos objetos que colisionan en función de la distancia de solapamiento d_{sol} entre sus eferas mínimas. Como árbol de referencia para ambos objetos se ha tomado el árbol correspondiente a la altura 16 de cada uno de ellos, lo que equivale a trabajar con una resolución de 0.05, donde la longitud del lado del cubo en el que se han realizado los experimentos es 5. La idea es generar de manera adaptativa las alturas de los árboles de forma que se obtenga un resultado comparable en términos del número de intersecciones detectadas al correspondiente con los árboles de referencia. Para ello se han generado medidas del número de colisiones que se detectan con árboles binarios cuyas esferas exteriores promedio tienen un diámetro menor que d_{sol}/n para $n = 3, 4, 5, 6$, donde d_{sol} es la distancia de solapamiento. En la tabla 3 se indican los resultados obtenidos en función del porcentage entre el número de intersecciones detectadas con la heurística y el número de intersecciones detectadas con el árbol de referencia. Como se observa en la tabla 3 prácticamente a partir de $n = 5$ se detecta casi el 95% del número total de intersecciones. Este resultado sugiere que la heurística es la adecuada para generar de manera adaptativa árboles binarios de esferas de objetos en entornos virtuales en los cuales se necesita verificar de las posibles interferencias entre objetos.

7 Conclusiones

En este trabajo se ha desarrollado un algoritmo de detección de intersecciones para objetos tridimensionales cuya geometría es la de un prisma recto con base un polígono generalizado. El algoritmo utiliza una representación jerárquica basada en esferas que consiste esencialmente en dos representaciones complementarias: i) una representación esférica de superficie y ii) una representación esférica del interior del objeto. Ambas representaciones se obtienen de un mismo arbol binario de esferas. La representación esférica de superficie es utilizada en la detección de la ausencia de intersecciones, mientras que la representación interior se aplica para detectar las intersecciones. La novedad del trabajo consiste en la utilización de las esferas de la representación interior para acelerar la detección de intersecciones en lugar de utilizar costosos test exactos de

Objeto-Objeto	SH	CH
Cubo-Cubo	102.6	24.5
Cubo-Caja	121.1	23.9
Cubo-Prisma 1	85.6	21.98
Cubo-Prisma 2	101.9	23.8
Caja-Caja	135.7	29.0
Caja-Prisma 1	109.21	25.59
Caja-Prisma 2	124.2	26.1
Prisma 1-Prisma 1	82.9	25.69
Prisma 1-Prisma 2	94.6	25.41
Prisma 2-Prisma 2	109.3	27.4

Tabla 2: Tiempos en la detección de intersecciones.

n	Cubo-Cubo	Prisma 1-Prisma 1
3	88.2%	74.7%
4	94.5%	88.8%
5	97.1%	94.7%
6	96.4%	97.8%

Tabla 3: Porcentages de detecciones de intersección calculadas.

intersección entre objetos. Varias medidas del desenpeño del detector de intersecciones han sido realizadas para verificar la eficiencia del mismo. El promedio obtenido en los tiempos de detección de intersecciones es del orden de entre 25 y 78 microsegundos para escenas virtuales consistentes en poliedros con un total de entre 16 y 204 vértices. Concluimos que el algoritmo desarrollado puede ser utilizado en tiempo real. Actualmente estamos extendiendo el detector de intersecciones a un detector de colisiones para realizar en tiempo real animaciones dinámicas con objetos virtuales.

Referencias

[1] del Pobil A.P., Pérez M. and Martínez B., A Practical Approach to Collision Detection between General Objects, Proc. IEEE Int. Conf. Robotics and Automation, (1996), pp. 779-784, Minneapolis, MN.

[2] Cameron, S. Enhancing gjk: computing minimum and penetration distance between convex polyhedra. Proceedings of International Conference on Robotics and Automation, Nagoya, 1997.

[3] Gilbert, E. G., Johnson, D. W., and Keerthi, S. S. A fast procedure for computing the distance between objects in three-dimensional space. IEEE Journal of Robotics and Automation, 4(2), (April 1988), pp. 193-203.

[4] Gottcharlk S., Lin M. and Manocha D., Obb-tree: A hierarchical structure for rapid interference detection. In Proc. of ACM Siggraph, (1996), pp. 171-180.

[5] Greenspan M. and Burtnyk N., Obstacle Count Independent Real-Time Collision Avoidance, International Conference on Robotics and Automation, (Abril 1996), pp. 1073-1079.

[6] Gupta K. and del Pobil A.P. (eds.), Practical Motion Planning in Robotics, John Wiley & Sons, 1998.

[7] Held M., Klosowski J. and Mitchell J., Evaluation of collision detection methods for virtual reality-fly-throughs. In Proc. Seventh Canadian Conf. Computer Geometry, (1995), pp. 205-210.

[8] Hubbard P.M. Interactive Collision Detection, Proceeding of the 1993 IEEE Symposium on Research Frontiers in Virtual Reality, (October 1993), pp. 24-31.

[9] Hubbard P.M. Approximating Polyhedra with Spheres for Time-Critical Collision Detection, ACM Trans. Graphics, vol. 15, no. 3, (1996), pp. 179-210.

[10] Hudson T., Lin M., Cohen J., Gottschalk S., and Manocha D. V-Collide: Accelerated collision detection for VRML. Proceeding of VRML Conference, (1997), pp. 119-125.

[11] Klosowski J., Held M., Mitchell J.S.B., Sowizral H., and Zikan K. Efficient collision detection using bounding volume hierarchies of k-dops. In Siggraph Visual Proceedings, (1996), pp. 151.

[12] Latombe J. C. Robot Motion Planning, Kluwer Academic Publishers

[13] Liu Y., Noborio J. and Arimoto S. Hierarquical sphere model and its application for cheking an interfernce between moving robots, In Proceeding of the IEEE International Workshop on Intelligent Robots and Systems, (1988), pp. 801-806.

[14] Martínez B., del Pobil A.P. and Pérez, M. Very fast collision detection for practical motion planning, Part I: The spatial representation. Proceeding of the 1998 IEEE International Conference on Robotics and Automation, (1998), pp. 624-629. Leuven, Belgium.

[15] Muñoz C., Arcila O., Bañón J.M. Nuevas Heurísticas para la Representación Eficiente de Objetos por Jerarquías de Esferas. XXVII Congreso LatinoAmericano en Informática, Mérida, Venezuela, (Septiembre 2001), pp. 28.

[16] O'Rourke J. and Badler, N., Decomposition of Three-Dimensional Objects into Spheres, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. PAMI-1, no. 3, (1979), pp. 295-305.

[17] Palmer I., Grimsdale, R. Collision Detection for Animation using Sphere-Trees. Computer Graphics Forum, 14(2), (1995), pp. 105-116.

[18] Pitt-Francis, J. and Featherstone, R., Automatic Generation of sphere hierarchies from CAD data Proceeding of the 1998 IEEE International Conference on Robotics and Automation, (1998), pp. 324-329, Leuven, Belgium.

[19] Tzafestas C. and Coiffet Ph., Real-Time Collision Detection using Spherical Octrees: Virtual Reality Application, IEEE International Workshop on Robot and Human Communication, (1996), Tsukuba, Japan.

[20] Zachmann G. Real-time and exact collision detection for interactive virtual prototyping, Proceeding of DECTC'97 ASME Design Engineering Technical Conferences, (1997), Sacramento, California.