CAPITULO 1. DETECCIÓN DE COLISIÓN

1.1 AMBIENTES VIRTUALES

Cuando un ambiente virtual contiene varios objetos que interactúan, la detección de colisiones es uno de los problemas fundamentales, ya que si no se presta atención especial en la interacción entre los objetos, se podrían originar estados no deseados entre ellos.

Figura 1.1 El mundo virtual esta lleno de objetos que interactúan [1]

En la figura 1.1, observamos que el robot al mover el bat que tiene en las manos éste penetra en el sillón, y con ello se genera un estado que no es valido en el mundo real. Un bat nunca penetra dentro del un sillón como lo muestra la figura.

Los objetos que forman parte de este ambiente deberán tener un comportamiento de acuerdo al objeto del mundo real que están representando. Detectar la proximidad y/o superposición entre objetos es importante para dar al ambiente un efecto más real. [PHI95]

Este problema es difícil, debido a que los objetos en el ambiente virtual, son modelados con formas simples (figuras geométricas como cuadrados, triángulos, etc.), en la mayoría de los casos la figura utilizada para la modelación de éstos es el triángulo.

En la figura 1.2, observamos un conejo, el cual es modelado con la figura geométrica del triángulo así como una tetera modelada con cuadrados.

[1] Imagen de Department of Computer Science University of North Carolina at Chapel Hill

Figura 1.2 Los objetos tienden a modelarse con figuras geométricas [2]

Un método para determinar que dos objetos modelados a través de triángulos están colisionando, consiste en verificar si cada uno de los triángulos de un objeto no intersecta a ninguno de los triángulos que forman el otro objeto; si la intersección de triángulos se presenta se puede asegurar que se ha producido una colisión entre los dos objetos. Si los objetos tienden a una forma más real, la malla de triángulos que los modela debe ser más fina, provocando un problema de cuello de botella ("Bottleneck"), debido a que el número de triángulos se incrementa así como el tiempo de verificación.

Dentro de un ambiente virtual o simulado existen varios objetos que interactúan y estas interacciones involucran que un objeto empuje, golpe o aplaste a otros, pero para cada uno de ellos su movimiento tiende a ser limitado por otros objetos en movimiento o estáticos, los cuales provocan que este pueda tener una colisión con ellos.

En un medio ambiente de simulación general con *N* objetos en movimiento y *M* objetos fijos. Cada uno de los *N* objetos en movimiento pueden chocar con los otros objetos en movimiento, así como con los objetos fijos. El mantener el control de $\binom{n}{2}$ + *NM* pares de objetos en cada espacio de tiempo puede llegar a quitar(o consumir) tiempo, mientras *N* y *M* se incrementan [LIN96], por lo que la detección de colisiones se convierte en el mayor cuello de botella ("Bottleneck"), para algunos simuladores interactivos.

Dentro de un ambiente real los objetos tienden a presentar determinados comportamientos, según sea la materia de la cual están compuestos. Dentro de un ambiente simulado los objetos deben presentar los mismos comportamientos de los objetos que representan. Por lo que el sistema de simulación requiere de un algoritmo de detección de colisiones.

La detección de colisión ha sido estudiada ampliamente, ya que es un factor importante en varias áreas, como la robótica, la realidad virtual, la simulación por computadora, la geometría computacional, el diseño asistido por computadora (CAD), graficas por computadora, ambientes virtuales, video juegos entre otras.

[2] Imagen de http://pgrafica.webideas4all.com/intropengl.html

Se han propuesto varias aproximaciones basadas en cajas envolventes, particiones de espacio, argumentos geométricos, métodos numéricos y métodos analíticos [HUD97, CAM91, HEL95, LIN93, GOT96].

La detección de colisión es el mayor cuello de botella en una simulación iterativa, asegurar que los objetos interactúan de una manera correcta es muy costoso.

Algunos investigadores en el área utilizan algoritmos de detección de colisión híbridos para agrupar el problema en varias fases.

La fase inicial conocida como la fase ancha (*broad phase*), cuyo objetivo es eliminar los objetos que se encuentran alejados unos de otros. Se ha propuesto diferentes técnicas para solucionar esto, ejemplo “Sweep & Prune” trabajadas por Cohen y Ponamgi, “global bounding volume tables” de Palmer y Grimsdale, y “overlap tables” de Wilson[COH95, PON95, WIL98, O´S99].

Habiendo determinado cuales objetos están potencialmente interactuando, los algoritmos híbridos usan un técnicas para limitar las regiones de los objetos las cuales están en contacto, a esto se le conoce como la fase angosta (*narrow phase*).

Para mejorar la eficiencia de tales algoritmos, las representaciones jerárquicas de los objetos se propusieron para localizar las áreas en donde la colisión realmente ocurre. Tales representaciones aproximan la superficie de un objeto en diferentes niveles de detalle. Como por ejemplo “Sphere tree” de Hubbard y Quinlan, “OBB-trees” (Oriented Bounding Boxes) de Gottschalk , “Shell trees” de Krishnan y jerarquías de “k-Dops” (Discrete Orientation Polytopes) de Klosowski[PHI95,QUI94, GOT96, KRI98, KLO97, O´S99].