TÍTULO (DESCRIPCIÓN CORTA DEL PROYECTO. ENTRE 8 Y 12 PALABRAS)

Daniel Hincapié
Universidad Eafit
Colombia
dahincapis@eafit.edu.co

Anthony García
Universidad Eafit
Colombia
agarciam@eafit.edu.co

Mauricio Toro
Universidad Eafit
Colombia
mtorobe@eafit.edu.co

1. INTRODUCCIÓN

Las abejas son una especie fundamental para la polinización, sin embargo, en la actualidad nos encontramos en dificultades para asegurar la supervivencia de esta importante especie. Hallamos diferentes factores contribuyentes a la desaparición de las abejas, tales como el uso de pesticidas, los cambios climáticos, la sobre-explotación que los humanos le dan, los depredadores, etc. Como una alternativa, se plantea el uso de abejas robóticas que se encargarían del desarrollo de la polinización.

2. PROBLEMA

Se nos presenta un problema a la hora de usar abejas robóticas, y es la posibilidad de colisiones entre estas, generando que los robots se destruyan y sean inutilizables. Es necesario resolver esta problemática para evitar la destrucción de los robots, cuestión que haría inefectiva la alternativa ante la desaparición de las abejas reales.

3. TRABAJOS RELACIONADOS

Encontramos problemas similares en los videojuegos, donde varios objetos pueden colisionar, para esto, se utilizan las estructuras de datos, entre las cuales podemos encontrar los Octrees, Quadtrees, tablas hash. En el caso de los Quadtrees, dividen los objetos que interactúan entre sí, en cuadrantes, evitando que estos colisionen.

3.1 Colisiones 2D en los videojuegos

En la realización de videojuegos 2D es común encontrar dos objetos que puedan tener una colisión, un ejemplo de esto, un perfecto ejemplo para esto lo podemos encontrar en el videojuego Brick Braker, en donde la bola colisiona con los ladrillos y debe rebotar nuevamente hacia la plataforma.

En este tipo de problemas, la solución que encontramos son los Quadtrees. Estos son un tipo de estructura de datos que se representan por medio de árboles, donde cada nodo se divide a su vez en otros cuatro nodos. Cada nodo se encarga de representar un cuadrante en nuestro plano 2D, y cada cuadrante contiene una cantidad de objetos, una vez un cuadrante está muy lleno y se hace difícil determinar las colisiones entre los objetos que lo componen, este se divide nuevamente en cuatro cuadrantes y reparte los objetos. Dicha actividad se repite hasta el punto en el que sea necesario.

3.2 Colisiones en videojuegos

En todos los videojuegos se presentan colisiones, ya que hay diferentes objetos que interactúan entre sí. Para evitar que un objeto que en la vida real no traspasaría a otro, traspase en un videojuego, se utilizan estructuras de datos.

Tenemos en este caso, los árboles octales, los cuales tienen una raíz y unas subdivisiones a partir de esta, impidiendo que el objeto de una las ramas del árbol colisione con un elemento de otra rama.

3.3 Falta de optimización en otras soluciones

Para evitar colisiones, se pueden usar soluciones como alguna de las ya mencionadas, por ejemplo los Quadtrees. Pero para programas o juegos que involucren una gran cantidad de colisiones a evitar, estas soluciones no van a ser óptimas.

Aquí aparecen las tablas Hash, y es que estos elementos nos ofrecen mayor optimización. Las tablas Hash nos permiten almacenar valores con y llaves de identificación. La clave se transforma en un hash y nos permite identificar rápidamente la posición que queremos encontrar en la tabla. Su principal ventaja es la velocidad con la que podemos trabajar, brindando resultados más óptimos.

REFERENCIAS

T. McDonald (2009, Septiembre 30). Spatial Hashing [en línea]. Disponible en: https://www.gamedev.net/articles/programming/general-and-gameplay-programming/spatial-hashing-r2697/