NON-COLLISION ALGORITHMS: A QUALITATIVE STUDY OF ALGORITHMS FOR COLLISION PREVENTION WITH ROBOTIC BEES

Juan Felipe Londoño
EAFIT University
Colombia
jflondonog@eafit.edu.co

Juan Pablo Giraldo Restrepo
EAFIT University
Colombia
jpgiraldor@eafit.edu.co

Mauricio Toro
EAFIT University
Colombia
mtorobe@eafit.edu.co

ABSTRACT

The future of pollination lies in the creation of mini drones that replace a bee's labor. In order for these drones to work in order they have to have certain characteristics and abilities, one of them being the skill to not crash into one another when being closer than 100 meters to each-other. Many problems rely on collision based algorithms as their solution. Such will be seen later on.

1. INTRODUCTION

We need bees. 70 of the top 100 human food crops, which produce 90% of the food we ingest, are pollinated by bees . Since 1947 to 2008 there has been a significant 60% reduction in bee hives. In the future we might not see bees anymore. Thanks to all of these factors and our dependence on bees, investigators are trying to develop artificial bees to pollinate our plants. Scientists such as Eijiro Miyako have been trying to develop insect sized drones to help or replace bees in their labor. All of this is not made easily. The drones must be able to work by themselves for them to have some weight in the pollination process. A very important factor for the robot bees to be autonomous is for them to be able to not crash between each-other which takes us to our problem. How can we make bees not crash into each-other when in distances that are under 100 meters? In this paper the reader will be able to see different solutions to the same problem using algorithms.

2. PROBLEM

As read in the preceding paragraph in the future bees may be extinct. For man kind to subsist alternatives such as autonomous pollinating bees must be made. In order for these mini robots to be independent they must be capable of not crashing with one another. For this to happen algorithms to control all of the bees and prevent crashes must be used.

3. RELATED WORK

3.1 Collision detection inside first person shooter video games

In most modern video games such as Call of Duty or similar, many collisions occur. When a bullet hits a player among other things. In order for the game to be able to know when this happens it uses algorithms. One of them is Axis Aligned Bounding Box also known as AABB which is a collision algorithm. The algorithm checks whether or not to items with a rectangular shape cause a collision on a 2D plane. Many video games use this algorithm as the first "test".

3.2 Algorithm development for the planning optimizing of routes of a mobile robot in an object based course

In some mobile robots one may also find collision related algorithms. In this case a robot wants to go from one place to another through an obstacle filled course. It uses Dijkstra's algorithm to find the shortest route possible, but it also uses another algorithm to avoid collisions. Here is where the Montecarlo algorithm comes in. It is in charge of comparing at every single point of the route, whether or not it is following the correct path. If the path is correct, the system will show the same value for the point in the path as well as the location of the robot. Otherwise both values will be different signaling a deviation in the route. In collision detection algorithms the program is constantly comparing two objects to see if they are in the same position, in order to conclude if there is a collision or not.

3.3 Development of an interface to simulate real-time organ-instrument interaction in a surgical procedure inside a teleoperation scenario

In medicine one may find procedures that contain collision detection. Everyday that passses we come closer to automating medical procedures as it is the case of a Student from Universidad Nacional de Colombia. He made a thesis based on a real-time simulation program for the interaction between instrument and organ inside a teleoperation scenario. The algorithm used for his problem is called GJK and is commonly used inside video games. It is based on collision detection between convex polytopes.

3.4 Development of an algorithm for collision detection hierarchy for the implementation of a laparoscopic surgery simulator

Another solution for another medical related problem is the implementation of an automated laparoscopic surgery. This project was made by Juan Sebastian Muñoz, Christian Andres Diaz and Helmuth Trefftz which are all part of EAFIT University. As mentioned before, medicine is in the search for automation or at-least simplifying processes. In this case the search for automation was with laparoscopic surgeries. Collision detection is needed in order for the laparoscopy to not collide with any vital organ. This solution was created based upon an algorithm made by Van Der Bergen who himself made a better AABB algorithm which is mentioned before. The idea is similar to Bergen's but instead of using an ascending update sequence, a middle level in the hierarchy is chosen as the star point.

REFERENCES

1. Eric Francisco Salinas. 2012. Desarrollo de algoritmos para la planeación y optimización de rutas de un robot móvil en ambientes con obstáculos. (February 2012). Retrieved February 25, 2018 from http://ri.uaq.mx/xmlui/handle/123456789/722

2. Jaime Andrés Castillo. 2011. DESARROLLO DE UN INTERFAZ PARA SIMULAR EN TIEMPO REAL LA INTERACCIÓN ÓRGANO-INSTRUMENTO EN UN PROCEDIMIENTO QUIRÚRGICO DENTRO DE UN AMBIENTE DE TELEOPERACIÓN. thesis.

3. Juan Sebastián Muñoz, Christian Andrés Diaz León, and Helmuth Trefftz Gómez. 2012. Desarrollo de un algoritmo de detección de colisiones jerárquica para la implementación de un simulador de cirugía laparoscópica. (June 2012). Retrieved February 25, 2018 from http://revistas.proeditio.com/iush/quid/article/view/63/63

4. Anon. Save the Bees. Retrieved February 25, 2018 from https://www.greenpeace.org/usa/sustainable-agriculture/save-the-bees/

5. Anon. 2D collision detection. Retrieved February 25, 2018 from https://developer.mozilla.org/en-US/docs/Games/Techniques/2D_collision_detection