Robotic bees: Data structures and algorithms for collision detection and prevention.
or
Robotic bees and how to stop them from "*bumper cart*-ing" into each other.

Vincent Alejandro Arcila
Universidad EAFIT
Colombia
vaarcilal@eafit.edu.co

Isabel Piedrahita
Universidad EAFIT
Colombia
ipiedrahiv@eafit.edu.co

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

Key words:

Author selected keywords: Spatial data structures, complexity, nearest neighbor, collision detection, efficient.

ACM CLASSIFICATION keywords: CCS → Information systems → Data management systems → Data structures → Data access methods → Proximity search

ABSTRACT

With the population of bees rapidly dropping in recent years, we are left with the task of finding ways in which to carry out tasks bees usually handle for us. One of such solutions might be the implementation of robotic bees, but with it arises the need to stop them from crashing into one another, resulting in a problem of collision detection in rigid yet constantly moving objects. **(To be completed in following deliverables)**

1. INTRODUCTION

In recent years the population of bees has been constantly decreasing . More than a billion of bees have dyed in Colombia alone in the span of the last four years[1], worldwide, bees are not doing any better. With the number of bees dwindling dramatically in the past 10 years it is time to consider the importance of bees in the ecosystem. Out of the 369,000 species of flowering plants, 90% depend solely on insect pollination[2], and taking into account that bees can visit anywhere between 50 and 1000 plants a day[3], it is undeniable that their role on this task is fundamental. Without bees not only would countless plants die, but the food chain would also be considerably damaged, since a sizable amount of primary producers would disappear. This urges us to find ways in which to cover up for the bees, which leads to solution such as the creation and deployment of robotic bees.

2. PROBLEM

A problem that will rise is how to get the robotic bees not to crash when they are pollinating. A small step to get with a solution is to find an efficient way to identify which bees are in risk to collide. In other words, alert bees when they are near to others.

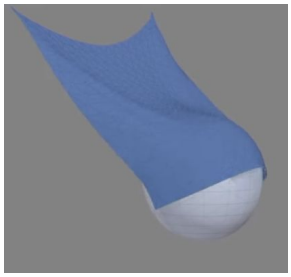
3. RELATED WORKS

3.1 Better collisions and faster cloth for Pixar's Coco[4]

While making Pixar's Coco animators ran into a complication, it was nearly impossible to get the cloth of clothes to sit properly on the huge cast of esqueletal characters, whose bones pinched the fabric causing what producers referred to as "Skeleton Wedgies". Because of this, their in-house cloth system, Fritz, had to implement a more efficient and easily systematized version of continuous collision detection.

In order to fix this, the Global Intersection Analysis, an algorithm that uses constant collision detection to identify when cloth has crossed over to the wrong side by analysing all the intersections between the digital mesh that composes the costumes. This was used to create a fast way to stop the cloth from going through

and getting stuck between the small geometries that make up Coco's skeletons.



3.2: Search and tracking algorithms for swarms of robots: A survey[5]
This paper serves as an introduction to any scientist interested on understanding ways to keep an eye on groups of robots and to make them work more efficiently. It covers various algorithmic related problems in order to give the reader an idea of what to have in mind when designing systems that involves groups of robots.

One of the problems that needs to be solved in software is the problem of target search and tracking, understanding a target as an object with which the robots need to interact directly in order to succeed on their task. Understanding how to handle with software all the different environments that the robots might have to face is not as simple as one might think. The number of targets may be unknown, or may vary with time. Also, how to manage robots so that they do not act on the same target simultaneously, known as the task allocation problem is a thing to have in mind. Targets might not be static, which has to be considered.

Cooperation and coordination among the robots is an essential aspect that has to be taken into account when designing swarm robots systems. An ideal system would be the one on which all robots are exchanging information, combining measurements from multiple positions to accurately determine the targets position, and get the best synchronization possible.

3.3: Cooperative Path Planning of Robot Swarm Based on ACO[6]
Ant Colony Algorithm is the abstraction of the way that ants find food. It is based on the fact that they secrete pheromones when walking, so the more pheromone is on a path the more ants have used it, meaning that there is likely to be food. The problem arises when a group of ants finds a deadlock area, meaning they have to go back to try a different path.

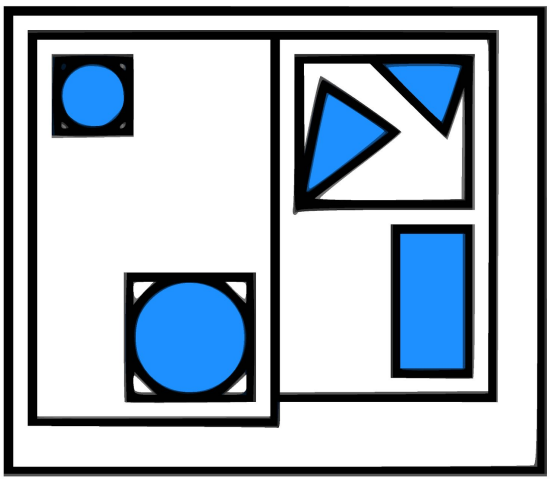
Can this algorithm be improved? Clearly, if no pheromone were on the path that leads to a deadlock ants would find food faster. The solution given in this paper states a barely obvious but hard to implement way to fix this issue. If the individual that found a deadlock removed the pheromone left on that path other individuals would not waste time on that path.

So, being robots instead of ants, we get a great increase on performance when a robot communicates to its peers that there is no point going through certain areas, due to the lack of targets on certain points of an open field.

3.4: Real-Time Robot Motion Planning in Dynamic Environments[7]
Navigation is one of the main fields collision detection is used on. As soon as a robot can move on its own, ithas to decide where to, and this is where collision detection comes in very handy. Although there has already been extensive study of collision detection, further study is very important, because current algorithms understand the world around them via a group of queries, instead of using geometrical analysis.

Because of this, the process of collision detection is usually divided in two phases.

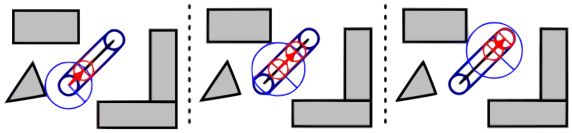
"Broad-Phase: Determines which geometric objects need to be checked for a certain query, ruling out distant objects based on bounding boxes or other partitionings of the space. Two data structures stand out throughout the broad-phase, namely, spatial hashing and spatial partitioning trees.



AABB-tree

Narrow-Phase: For the objects that overlap in the broad-phase, a check is performed between the query and a single primitive object that represents an obstacle. Often objects correspond to rigid bodies."

In the narrow-phase some very useful algorithms stand out when we take into account that most of the structure is described by primitive objects such as spheres. These algorithms include checking a point q and radius r for collision, calculating the distance from q to the nearest point on an obstacle and Calculating the nearest point p on the obstacle to a query point q.



Example swept circle obstacle check using only distance queries.

3.5: Optimized Spatial Hashing for Collision Detection of Deformable Objects[8]

In this paper the structure presentes a data structure that seems 'perfect' to work with GPU. The basic principle behind this structure is to precompute as much as possible, so is more likely to not have collisions in it. It also is spatial coherent, meaning that the objects that are near in the data structure are also near in the three-dimensional space, resulting in efficient access and comparison between objects inside the table.

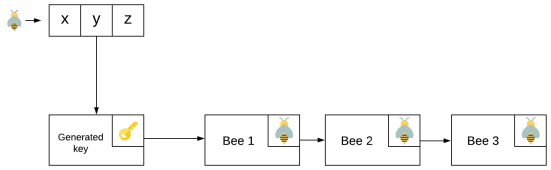
5. SPATIAL HASHING

Spatial hashes are a hash table in which each key corresponds to a 3D coordinate and the value is a vector of objects, in this particular case bees, that are located in said space.

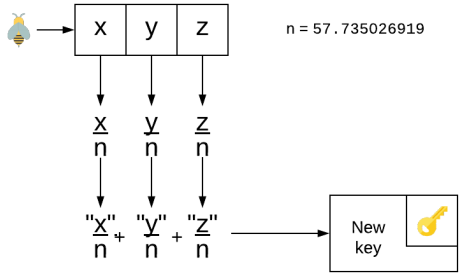
The basic principle is to split space into an infinite number of cells, where each cell can contain an arbitrary number of items. Additionally, any cell that does not contain an object, will not be created, saving space in memory.

5.1: Operations of the data structure

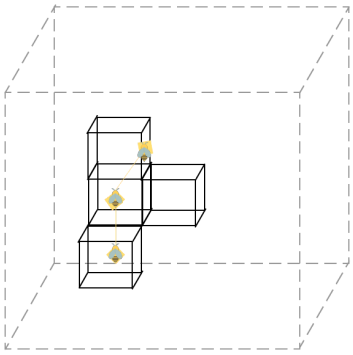
Parse file: This operation will receive the information regarding a particular group of bees from a .txt file, it will use this information to generate a key, and proceed to insert the bee in the position of the hash table that corresponds to said key. If it is the case that a bee has already occupied this space then the bee will be added to a linked list of bees.



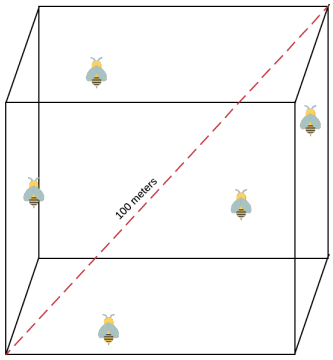
Find cube key: This operation will take the coordinates in meters of each bee and divided by the number required to create a 100 meter diagonal. It will then concatenate them and use this final string as a key.



Find for unique bee: This method will take any bee that is alone on a cube it will search all of the adjacent cubes in the spatial hash and compare the distance between all of the bees contained in them and the current bee.



Find in cube: This method will use a boolean flag to add bees in the same cube to a linked list of bees. As the cube has a diagonal of 100 meters, then it is clear that all of these bees are at risk of collision. Therefore they can be written onto the output file.



5.2: Design criteria for the data structure

Our data structure is required to represent the space intuitively, in order to improve the complexity of the *search collisions* method by decreasing the amount of operations performed to detect these events for a single point in space. Considering this, we decided spatial hashing would be a simple and efficient way of accurately representing our data, with an added bonus the KD-tree did not have, any given pair of bees that are close on real space, will also be close on the hash table.

5.3: Complexity analysis

Method	**Complejidad**
Insert	O(n)
Find cube key	O(1)
Find unique bee	O(n)
Find in cube	O(1)

5.4: Execution time and Memory used

In order to keep the results of the charts relevant we chose to only show two of the above described methods.

Parse file	**Memory**	**Time**
Parse file		
Find cube key		
Find unique bee		
Find in cube		

Find collision	**Memory**	**Time**
Parse file		
Find cube key		
Find unique bee		
Find in cube		

5.5: Result analysis

6. CONCLUSION

We conclude that spatial hashing is an efficient and logically simple data structure capable of achieving the goal of this paper, but the way in which keys were handled in order to ensure the proper

6.1: Future work

woIn future work we would like to implement a similar data structure capable of handling data of moving bees, rather than a screenshot of their current position. Furthermore we would like to make our nearest neighbor algorithm more memory efficient.

ACKNOWLEDGEMENTS

We would like to thank professors Pineda and Lalinde, from the Apolo supercomputation center.

Addiotionally, we would like to thank the teacher's assistants.

7. TEAMWORK

alIn order to see our gradual progress from deliveries one and two of this project visit: https://drive.google.com/file/d/1Wbb43t35TPKfuSZwQhD-VBcz06Ti7t8U/view?usp=sharing

REFERENCES

[1] RCN radio, Carlos Brand (2018) Más de mil millones de abejas han muerto en Colombia en los últimos tres años. (16, February 2018). Retrieved February 15, 2019 from https://www.rcnradio.com/recomendado-del-editor/mas-de-mil-millones-de-abejas-han-muerto-en-colombia-en-los-ultimos-tres

[2] BBC News, Rebecca Morelle (2016) Kew report makes new tally for number of world's plants. (10,May 2016). Retrieved February 15, 2019 from https://www.bbc.com/news/science-environment-36230858

[3] Nicola Bradbear BEES AND THEIR ROLE IN FOREST LIVELIHOODS: A guide to the services provided by bees and the sustainable harvesting, processing and marketing of their products. FAO, Rome 2009, 13-16.

[4] David Eberle. 2018. Better collisions and faster cloth for Pixar's Coco. In ACM SIGGRAPH 2018 Talks (SIGGRAPH '18). ACM, New York, NY, USA, Article 8, 2 pages. DOI: https://doi-org.ezproxy.eafit.edu.co/10.1145/3214745.3214801

[5] Madhubhashi Senanayake, Ilankaikone Senthooran, Jan Carlo Barca, Hoam Chung, Joarder Kamruzzaman, Manzur Murshed. 2015. Search and tracking algorithms for swarms of robots: A survey. *Robotics and Autonomous Systems 75, B (2016) 422-434.* DOI:https://doi.org/10.1016/j.robot.2015.08.010.

[6] Li Yong, Li Yu, Guo Yipei,,Cai Kejie. 2017. *Cooperative Path Planning of Robot Swarm Based on ACO*. Key Laboratory of Industrial Internet of Things and Intelligent Instrument, Ministry of Education. Industrial IoT Collaborative Innovation Center Chongqing Education Commission. Chongqing University of Posts and Telecommunications, Nanshan Scenic, 渝.

[7] James Robert Bruce. 2006. Real-Time Motion Planning and Safe Navigation in Dynamic Multi-Robot Environments. Ph.D. thesis Dissertation. Carnegie Mellon University Pittsburgh, PA 15213.

[8] Sylvain Lefebvre, Hugues Hoppe. ND. Perfect Spatial Hashing. Microsoft Research.