Advanced Science and Technology Letters
Vol.133 (Information Technology and Computer Science 2016), pp.223-228
http://dx.doi.org/10.14257/astl.2016.133.42

Autonomous Vehicles Should Not Collide Carelessly

Yair Wiseman and Ilan Grinberg[1]

[1] Computer Science Department, Bar-Ilan University, Ramat-Gan 52900, Israel
wiseman@cs.biu.ac.il

Abstract. Occasionally an autonomous vehicle unfortunately recognizes that an unavoidable accident is going to happen; yet the autonomous vehicle still has a number of possible options of accidents and it is still able to choose what the least harmful option is. A system for a real-time assessment of vehicle accident potential damages is presented in this paper. This system improves the vehicle's embedded computer ability to choose the least harmful decision.

Keywords: Autonomous Vehicle, Car Accident, Embedded Real-Time System.

1 Introduction

The "Trolley Problem"[1] is a well-known dilemma. It is relevant to the autonomous vehicle industry when an autonomous vehicle should decide and select between several damaging actions in the course of an inescapable accident. Such morals questions like "Who should die the driver of the car or a pedestrian in the vicinity of the car?" have been debated by many philosophers, religions and law makers. In this paper we do not intend to find the answers for these moral dilemmas. We would like to focus in the eminent subject of the passenger safety [2]; accordingly, we would strive for giving techniques for assessing the potential damages that possibly will happen in each course of action.

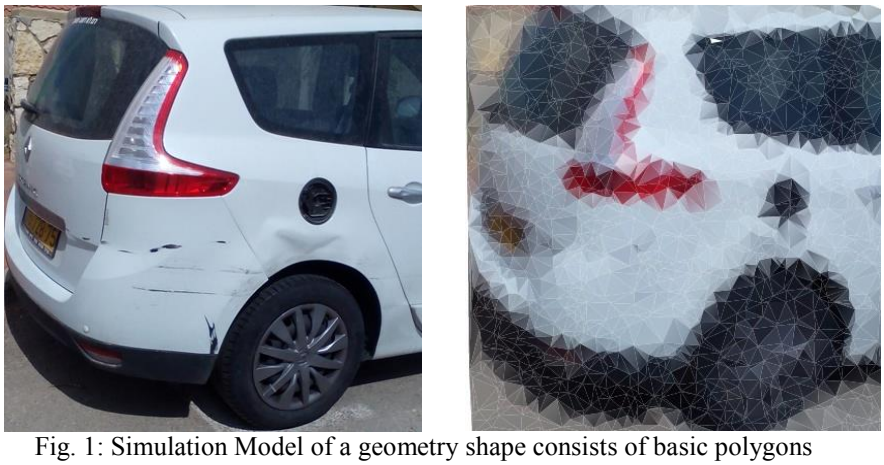

Fig. 1: Simulation Model of a geometry shape consists of basic polygons

ISSN: 2287-1233 ASTL

One of the most widespread techniques for effectively employing computational geometry functions is creating an intelligent simulation model for each geometry structure consists of simple polygons. Fig. 1 is an example for such a simulation model for a vehicle after a crash. The right picture is the original picture of the damaged vehicle. The left picture is the simulation model consists of simple polygons.

Spatial Data Structures are employed in two main approaches. The first approach is diminishing the number of intersection detections of vehicle models in a given scenario [3,4]. For n structures, there will be $O(n^2)$ possible structures that possibly will be intersected. This number is clearly too high; therefore diminishing the number of intersection checks achieved by Spatial Data Structures is imperative.

The second approach is diminishing the number of intersection checks of pair of simple polygons in a crash probe of two vehicle models. In this approach, the Spatial Data Structures are generated in a preprocessing step and remain static because the vehicles are rigid and their models do not changed.

Spatial Data Structures are often employed for Space Partitioning [5] and Bounding Volumes [6]. Space Partitioning is a sub-partitioning of a space into convex regions called cells. Each cell keeps a list of objects that it contains. Using these structures, a computer can sifted out numerous pairs of objects.

Bounding Volume is generated by a split of an object set into several subsets and finding for each one of the subsets tight bounding volume; so as a result when the computer checks intersections of each of the subsets, it will straightforwardly sifted out these subsets because it will only have to find out which bounding volumes are not overlapping. Hardware-software codesign [7] can perform this even better.

Some research have been conducted on approaches of representing Bounding Volumes like Bounding Spheres [8], K-DOPs - Discrete orientation polytopes [9], OBB - Oriented Bounding Boxes [10], AABB - Axis Aligned Bounding Boxes [11] and Hierarchical Spherical Distance Fields [12].

We will employ the most widespread approach, the AABB approach. In this approach the bounding volume is denoted by minimum and maximum values of the vehicle model in each one of the axes. The disadvantage of the AABB approach is that its representation is more memory consuming than the "Bounding Sphere" approach; however, nowadays memory chips are much larger and much more inexpensive and furthermore AABB has an important advantage – its objects can more tightly enclose the vehicle model than Bounding Spheres can, which will generate less intersection checks.

Another advantage of AABB is the quick construction of bounding volumes [13]. This advantage is very important in a case of an autonomous vehicle accident when the vehicle's computer does not have much time to make its decisions. The computer just needs to check each element of the basic elements that the bounding volume consists of and projecting it on each of the axes. After that, just finding the minimum value and the maximum value for each axis and the construction is done.

In view of that, we employed the AABB approach. The creation of the bounding box tree has been recursive. First, the computer computes a bounding box for the set of the remaining triangles. Then, the computer splits the set of the triangles into two sub-meshs. At last, the computer executes the recursive process on the two new split sub-meshs.

2 Triangles Split Algorithm

Each triangular mesh with a corresponding bounding box can be split into two sub-meshes. The Triangles Split Algorithm is described herein below:

- Let min_sum be the maximum value that a float variable can represent.
- For each of the box axes:
  - Select a positive direction for the axis.
- For each triangle
  - Find the maximal valued vertex on the projected axis.
- Sort the triangles by their maximal vertex value.
- For each triangle from the minimum to the maximum:
  - Tag the triangle as a "split triangle" (This tag indicates that the first sub-mesh will contain the triangles from the minimal to the split triangle; whereas the second sub-mesh will contain the rest of the triangles).
  - Calculate the sum of the relative segments of the two sub-meshes. (A relative segment is the length of the projection of a sub-mesh onto the box axis divided by the original mesh projection length).
  - If the relative segments sum is less than the min_sum:
    - Let min_sum be the new relative segments.
    - Tag the current axis as the split axis.
    - Tag the current triangle index as the split index.
  - Split the triangles according to the latest split axis and the latest split triangle index.

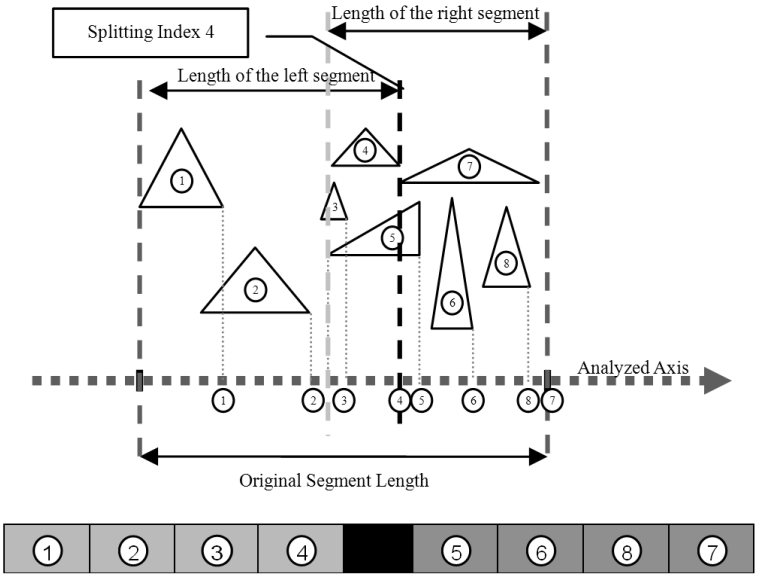

Fig. 2: triangles' set arranged from left to right on the projecting axis with splitting index 4

The incentive of the algorithm is guaranteeing that there is the smallest possible overlapping between the two sub-meshes' bounding boxes.

Fig. 2 and Fig. 3 show a paradigm of two different split indices. Fig. 2 shows a possible split at triangle index 4. Such a split will generate a larger overlapping between the two divided segments than a split in triangle index 2. Fig. 3 shows this possible split in triangle index 2 and actually this explains why the algorithm would select triangle index 2 to be the split triangle if this was the case.

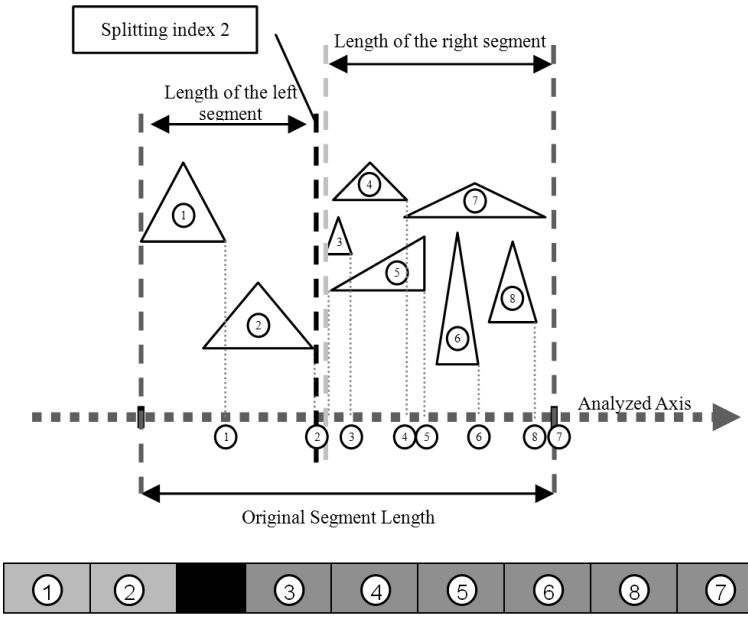

Fig. 3: triangles' set arranged from left to right on the projecting axis with splitting index 2

3 Conclusions

The up and coming autonomous vehicles strengthened the research of embedded vehicular and transportation computing systems [14,15,16]. The autonomous vehicles encouraged many researchers to revisit well-known subjects of computer science [17,18,19].

Real time computer graphics implementations like 3D game engine frequently employ the primitive intersection approach [20]. In this paper we have shown how to adapt this common approach into a specific assessment implementation for potential damages of vehicle accident. The goal of this implementation is an automatic decision maker for autonomous vehicles that can choose in an inescapable accident situation, which kind of accident is the least damaging one.

References

1. J. J. Thomson, "Killing, letting die, and the trolley problem", Ethical Theory - An Anthology, Second Edition, John Wiley & Sons, Inc. Publication, pp. 204-217, 1976.
2. Y. Wiseman and Y. Giat, "Multi-modal passenger security in Israel Multimodal Security in Passenger and Freight Transportation: Frameworks and Policy Applications, Edward Elgar Publishing Limited, Chapter 16, pp. 246-260, 2016.
3. Y. Wiseman and E. Fredj, "Contour Extraction of Compressed JPEG Images", Journal of Graphic Tools, Vol. 6, No. 3, pp. 37-43, 2001.
4. E. Fredj and Y. Wiseman, "An O(n) Algorithm for Edge Detection in Photos Compressed by JPEG Format", Proceedings of IASTED International Conference on Signal and Image Processing SIP-2001, Honolulu, Hawaii, pp. 304-308, 2001.
5. A. V. Husselmann, A. Hawick, "Spatial Data Structures, Sorting and GPU Parallelism for Situated-agent Simulation and Visualisation", In Proceedings of International Conference on Modelling, Simulation and Visualization Methods, pp. 14-20, Las Vegas, USA 2012.
6. C. Stein, M. Limper, and A. Kuijper, "Spatial data structures for accelerated 3D visibility computation to enable large model visualization on the web" In Proceedings of the 19th International ACM Conference on 3D Web Technologies, pp. 53-61, 2014.
7. Y. Wiseman, "A Pipeline Chip for Quasi Arithmetic Coding", IEICE Journal - Transactions Fundamentals, Tokyo, Japan, Vol. E84-A No.4, pp. 1034-1041, 2001.
8. S. Pabst, A. Koch and W. Straßer, "Fast and scalable cpu/gpu collision detection for rigid and deformable surfaces", In Computer Graphics Forum, vol. 29(5), pp. 1605-1612, 2010.
9. T. Ning, L. J., Wang, B. Li, "A Novel Method Based on K_DOPs and Hybrid Bounding Box to Optimize Collision Detection", Journal of Convergence Information Technology, Vol. 7, No. 12, pp. 389-397, 2012.
10. J. W. Chang, W. Wang and M. S. Kim. "Efficient collision detection using a dual OBB-sphere bounding volume hierarchy", Computer-Aided Design, Vol. 42(1), pp. 50-57, 2010.
11. I. Grinberg and Y. Wiseman, "Scalable Parallel Collision Detection Simulation", In Proceedings of Signal and Image Processing, Honolulu, Hawaii, pp. 380-385, 2007.
12. R. Weller, "New Geometric Data Structures for Collision Detection and Haptics", Springer Science & Business Media, 2013.
13. I. Grinberg and Y. Wiseman, "Scalable Parallel Simulator for Vehicular Collision Detection", International Journal of Vehicle Systems Modelling and Testing, Vol. 8, No. 2, pp. 119-144, 2013.
14. R. Ben Yehuda and Y. Wiseman, "The Offline Scheduler for Embedded Vehicular Systems", International Journal of Vehicle Information and Communication Systems, Vol. 3, No. 1, pp. 44-57, 2013.
15. R. Ben Yehuda and Y. Wiseman, "The Offline Scheduler for Embedded Transportation Systems" In Proceedings of IEEE Conference on Industrial Electronics (IEEE ICIT-2011), Auburn, Alabama, pp. 449-454, 2011.
16. P. Weisberg and Y. Wiseman, "Efficient Memory Control for Avionics and Embedded Systems", International Journal of Embedded Systems, Vol. 5(4), pp. 225-238, 2013.
17. Y. Wiseman, "Take a Picture of Your Tire!", Proceedings of IEEE Conference on Vehicular Electronics and Safety (IEEE ICVES-2010) Qingdao, ShanDong, China, pp. 151-156, 2010.
18. Y. Wiseman, "The Effectiveness of JPEG Images Produced By a Standard Digital Camera to Detect Damaged Tyres", World Review of Intermodal Transportation Research, Vol. 4, No. 1, pp. 23-36, 2013.
19. Y. Wiseman, "Camera That Takes Pictures of Aircraft and Ground Vehicle Tires Can Save Lives", Journal of Electronic Imaging, Vol. 22, No. 4, 041104, 2013.

20. D. H. Eberly, "3D Game Engine Design: A Practical Approach to Real-Time Computer Graphics", CRC Press, 2006.

