When an Inescapable Accident of Autonomous Vehicles is Looming

Yair Wiseman and Ilan Grinberg

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

Abstract

A system for a real-time estimation of vehicle accident potential damages is proposed with the aim of facilitating autonomous vehicle's embedded computer system to come to the least harmful possible decision if it realizes that an accident is unavoidable. The system analyzes all the possible crashes so the autonomous vehicle will make a decision what the least destructive option is.

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

1. Introduction

Unfortunately, there is no obvious way to encode human ethics in computers [1,2]; however, ethical choices on no-win situations similar to the well-known "Trolley Problem"[3] can take place in circumstances where an autonomous vehicle should decide and select between several damaging actions in the course of an inescapable crash. Such ethical alternatives 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 ethical decisions. We would like to focus in the eminent subject of the passenger safety [4]; accordingly, we would strive for giving techniques for assessing the potential damages that possibly will happen in each course of action [5,6].

Figure 1. Simulation Model of a Geometry Shape Consists of Basic Polygons

Figure 2. Original Image of the Car in Figure 1

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. Figure 1 is an example for such a simulation model for a vehicle after a crash. Figure 2 is the original picture of the damaged vehicle; whereas Figure 1 is the simulation model consists of simple polygons. The damaged parts are highlighted.

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 [7,8]. 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 [9] and Bounding Volumes [10]. 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 [11] can perform this even better.

Some research have been conducted on approaches of representing Bounding Volumes like Bounding Spheres [12], K-DOPs - Discrete orientation polytopes [13], OBB - Oriented Bounding Boxes [14], AABB - Axis Aligned Bounding Boxes [15] and Hierarchical Spherical Distance Fields [16].

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 [17]. 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. Bounding Volume Tree Generation

Bounding volume hierarchies [18] are data structure of a tree whose leaves are the basic elements of the geometry. Each set of sub-tree' leaves represents a bounding box. Sibling roots of sub-trees can overlap if their representing bounding volumes are overlapping.

Using bounding box hierarchies has two main advantages:

- Fast response for query of intersection check
- Linear storage usage with respect to the number of elements constructing the geometry.

The main disadvantage of using bounding box hierarchies is the long time it takes to construct the representing tree of the geometry and the updates that are required when using non-rigid solids. This explains why the use of bounding volume hierarchies is common mostly when using rigid geometries when the representing tree is produces only once as a pre-processing step.

The collision check between two geometric models is done recursively for each two nodes taken from each of the geometries trees, beginning with the roots.

The entire time of collision detection between two geometric models, which are represented by two bounding volume hierarchies, can be calculated by the following formula:

$T_{total}=N_b C_b + N_p C_p$

Where,

T_{total} – The entire time for intersection check between the two geometric models.

N_b – Number of bounding volume pairs checked for an intersection.

C_b – The time of one intersection check between a pair of bounding volumes.

N_p – Number of primitive polygon pairs checked for an intersection.

C_p – The time for an intersection check between a pair of primitive polygons

The parameters affected by the bounding volume type are N_b, N_p, and C_b. A tight-fitting bounding volume type, such as OBB, will cause N_b and N_p to be low, but has a pretty higher C_b, whereas AABB will produce more checks, but the value of C_b will be lower.

Our methodology to constructing bounding volume tree has been recursive. The process has been split into three major steps:

- Create a bounding volume for the set of remained triangles.
- Split the set of triangles into two submeshs.
- Execute the recursive process on the two new split sub-meshes.

The two new split sub-meshes of triangles represent the child nodes of the triangles' initial group node that contains the two sub-meshes. If a sub-mesh contains at least two triangles, then the process will be rerun on that sub-mesh.

The creation of the bounding volume algorithms and the triangle split algorithms have an important effect on the bounding volume tree creation algorithm and its performance. We use "Fitting points with Gaussian distribution" to create the bounding volumes as described in [19].
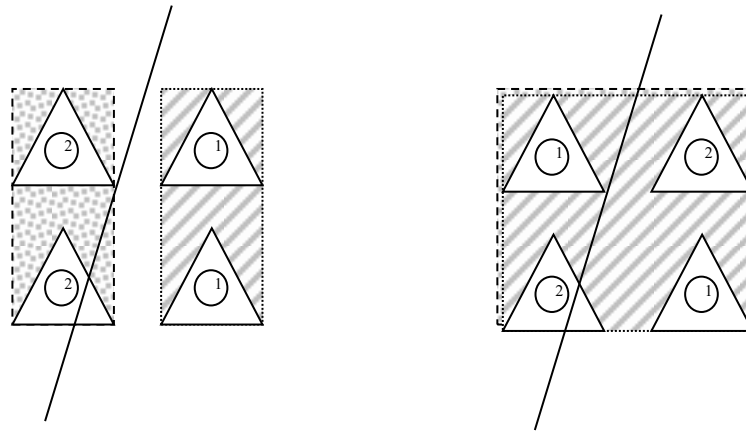

Figure 3. Example of Triangle Split

The motivation for the split of the triangles into two sub-meshes is creating bounding volumes with minimal dimensions for the sub-meshes.

Figure 3 depicts an example of four triangles split in two different ways. The number within each triangle represents the sub-mesh the triangle belongs to after the split. This figure demonstrates that a hierarchical intersection checking with a specific segment may create less triangle intersection checks in the left side of the figure because the bounded volume has a smaller dimension. This feature was the major motivation to use the split algorithm described in the next section.
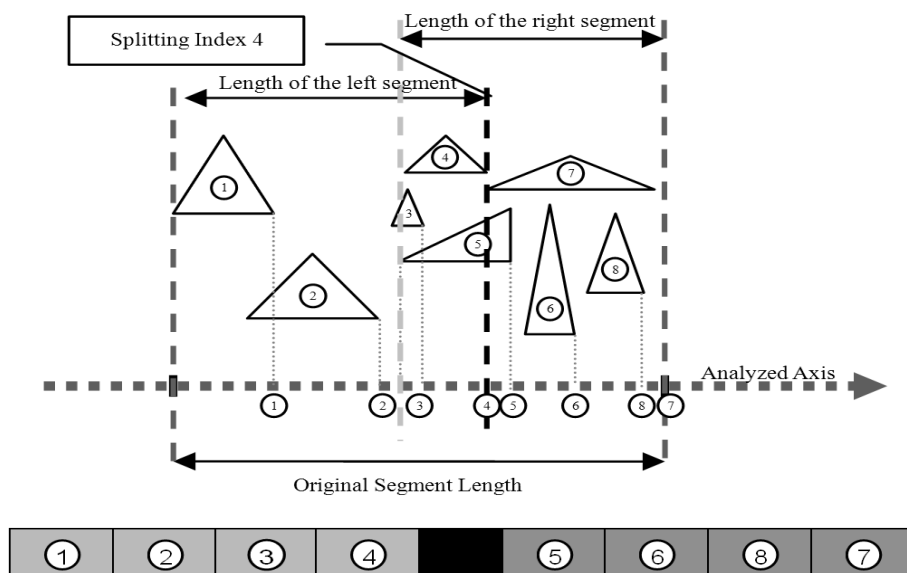
Figure 4. Triangles' Set Arranged from Left to Right On the Projecting Axis with Splitting Index 4
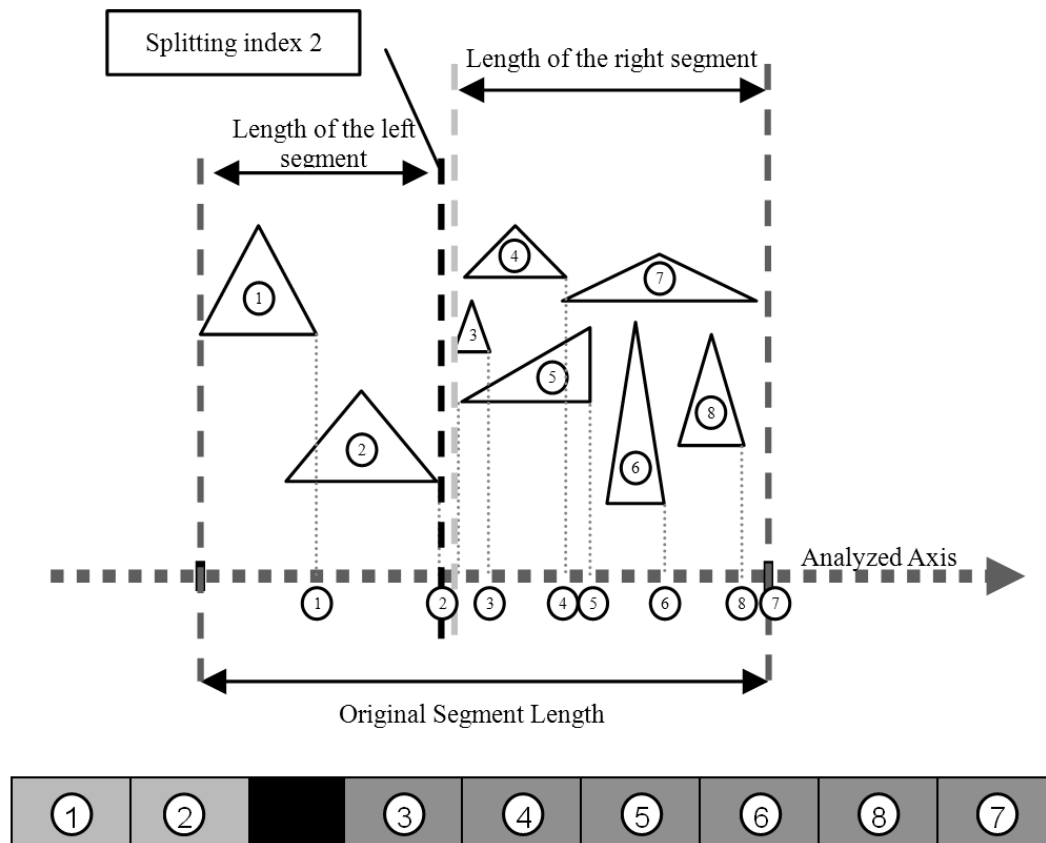
Figure 5. Triangles' Set Arranged from Left to Right On the Projecting Axis with Splitting Index 2

3. 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.

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

Figure 4 and Figure 5 show a paradigm of two different split indices. Figure 4 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; whereas Figure 5 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.

4. Results

We used Intel® Pentium® Processor N3540 which is a very common quad-core processor with 2.16GZ. We aimed at gauging the efficiency of the triangle scheme we have used in this paper.

The triangles were processed by the four cores of the processor. We compared the following schemes:

- **Best Match** - Select the core with the highest number of similar geometry parts.
- **Random Match** - For each check, a random unclaimed core will be selected.
- **Lowest Match** - Select the core with the lowest number of similar geometry parts.
- **Best Match-Load** - The motivation of this scheme is not to load cores with many checks. Loaded cores should not be selected to make the next checks if a less loaded core is available, even if the less loaded core's geometry is less similar. The load on a core is calculated by dividing the buffered checks in the core by its buffer maximal size. We took into consideration both the load and the geometry similarity to the core's checks with the aim of obtaining the best possible performance.

There are several factors that we should take into account when distribution depth of the bounding volumes is selected. If a small depth is selected, the geometry will be split into large overlapping bounding volumes, which will cause a retrieval of many node collision pairs. On the other hand, if a large depth is selected, the main process will waste more time in analyzing the first step of the collision detection and as a result it may possibly generate a bottleneck.
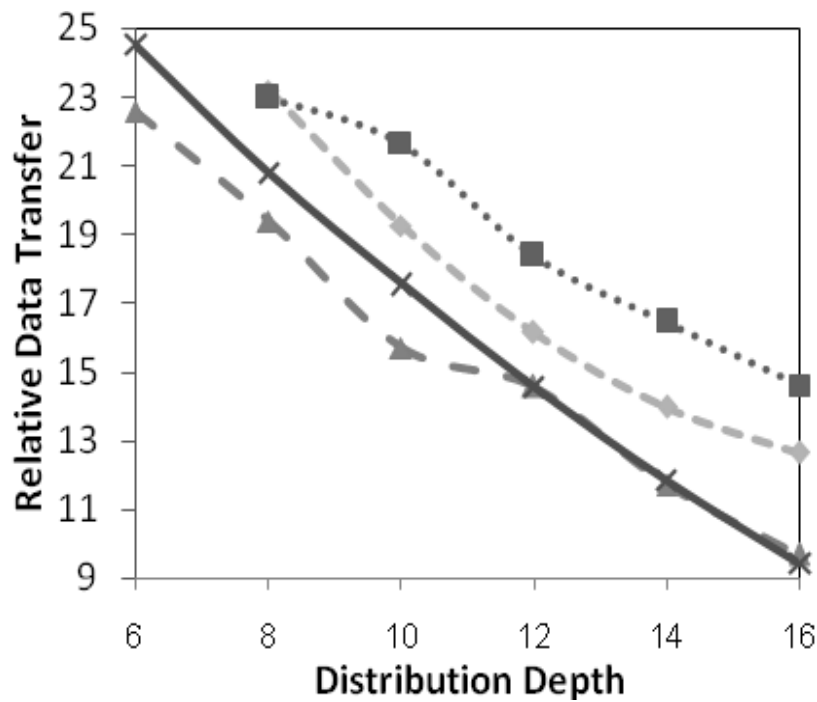
Figure 6. Another Simulation Model of a Geometry Shape Consists of Basic Polygons

We have examined the influence of several depths on the performance. We have used the image of the car in Figure 1 and the image of the car Figure 6.
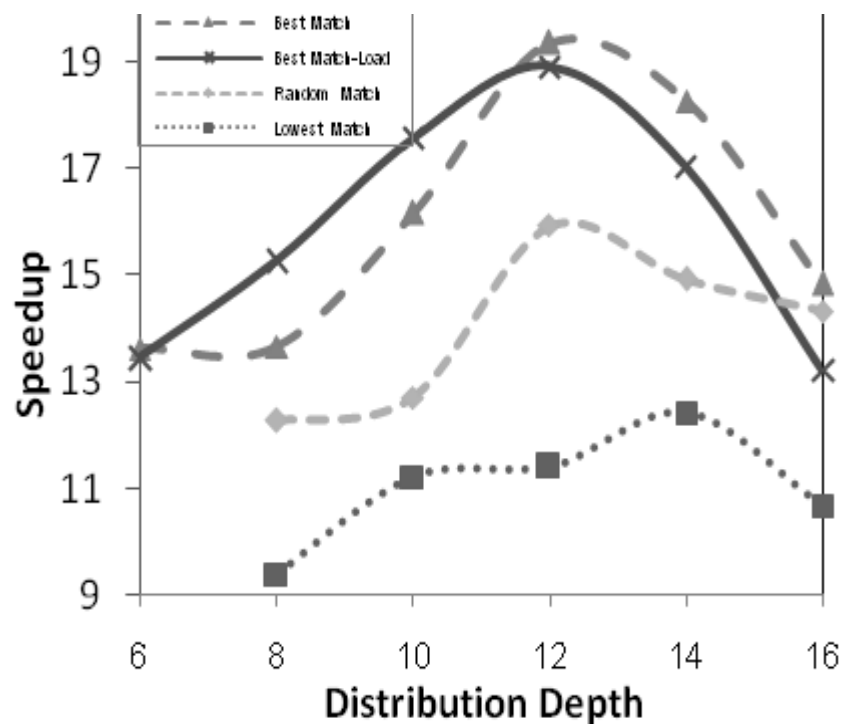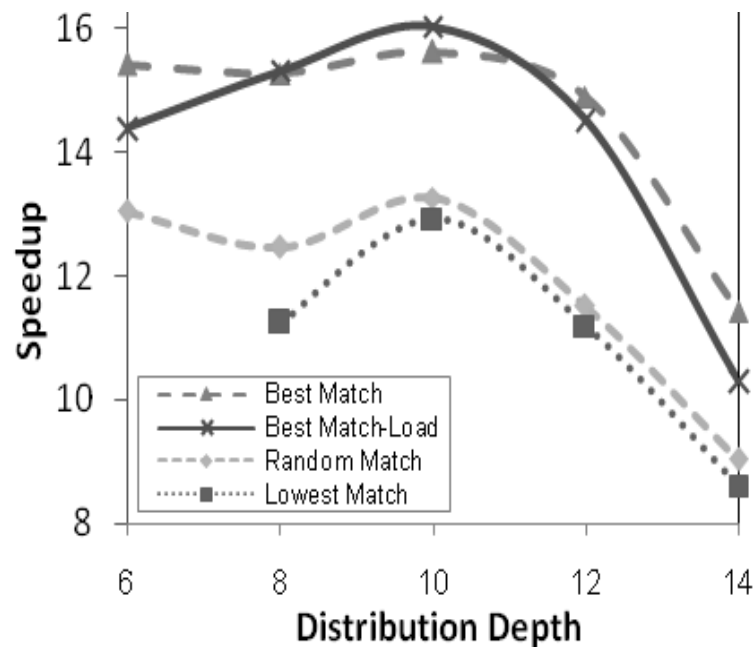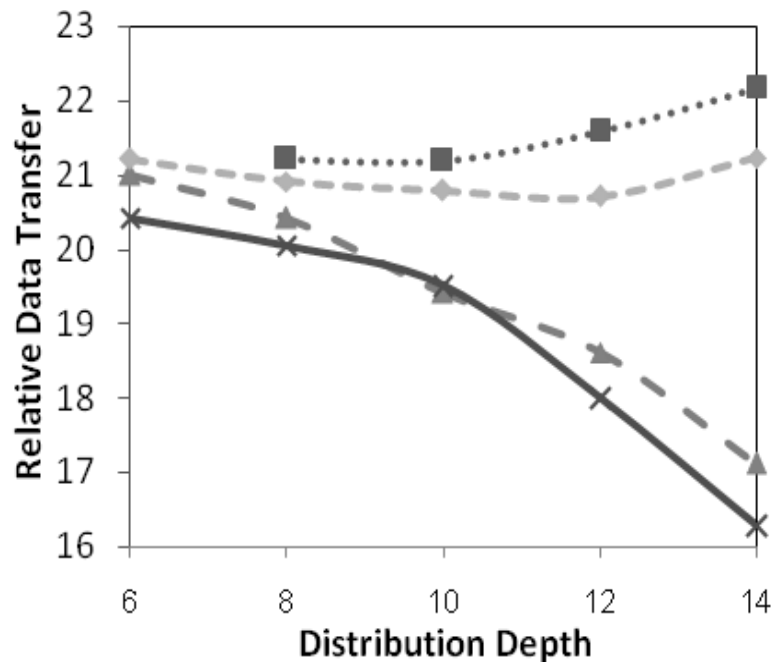
Figure 7. Distribution Depth Analysis of Speedup and Relative Data Transfer of the Car in Figure 1

The results are shown in Figure 7 and Figure 8. It can be clearly seen that the Best Match scheme gives better performance, both in speedup and relative data transfer.

Figure 7 and Figure 8 show in addition that different geometry models have different optimal distribution depth, specifically, In Figure 7 the optimal distribution depth is 12 and in Figure 8 the optimal distribution depth is 10.

Figure 8. Distribution Depth Analysis of Speedup and Relative Data Transfer of the Car in Figure 6

It can be also concluded from Figure 7 and Figure 8 that if the given geometry model is bigger, the relative performance of the Best Match algorithm will be better.

We can see in Figure 7 and Figure 8 trimmed lines in the low distribution depth of Lowest Match and Random Match algorithms. This missing information has been ensued as a result of a lack of memory.

At the initial stage, when the first jobs are sent, all the calculations wait for relocation to other cores in the processor. If a memory wasteful algorithm like Lowest Match or Random Match is used, the processor can quickly run out of memory and will be unable to handle the task.

The buffer size of the cores has a noticeable effect on the performance of the algorithms. The buffer gives a core the option of collecting several jobs and sending them en masse to another core. Therefore, the main process has more time for setting up tasks for other cores.

5. Conclusions

Autonomous vehicles are an emerging concept that gave a boost to the embedded vehicular and transportation computing systems [20, 21, 22]. The autonomous vehicles motivated many researchers to revisit well-known subjects of computer science [23, 24, 25].

Primitive intersection is a well-known technique for real time computer graphics implementations like 3D game engine [26]. We suggested in this paper how to adapt this very general concept into a specific assessment tool for potential vehicle crash damage. Such a tool's aim is an automatic decision maker for autonomous vehicles that will decide in an inescapable accident scenario, which sort of accident is the least harmful.

References

[1] N. Goodall, "Ethical Decision Making during Automated Vehicle Crashes", Transportation Research Record, Journal of the Transportation Research Board, The National Academies of Sciences, Engineering, and Medicine, vol. 2424, **(2014)**, pp. 58-65.

[2] N. Goodall, "Machine Ethics and Automated Vehicles", Road Vehicle Automation, Springer, **(2014)**, pp. 93-102.

[3] J. J. Thomson, "Killing, letting die, and the trolley problem", Ethical Theory - An Anthology, Second Edition, John Wiley & Sons, Inc. Publication, **(1976)**, pp. 204-217.

[4] 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, **(2016)**, pp. 246-260.

[5] Y. Wiseman, "Circumspectly Crash of Autonomous Vehicles", To appear in IEEE International Conference on Electro Information Technology, Grand Forks, North Dakota, USA, May 19-21, **(2016)**.

[6] Y. Wiseman, "Autonomous Vehicles Should Not Collide Carelessly", To appear in Advanced Science and Technology Letters, **(2016)**.

[7] Y. Wiseman and E. Fredj, "Contour Extraction of Compressed JPEG Images", Journal of Graphic Tools, vol. 6, no. 3, **(2001)**, pp. 37-43.

[8] 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, **(2001)**, pp. 304-308.

[9] A. V. Husselmann and 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, Las Vegas, USA, **(2012)**.

[10] 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, **(2014)**, pp. 53-61.

[11] Y. Wiseman, "A Pipeline Chip for Quasi Arithmetic Coding", IEICE Journal - Transactions Fundamentals, Tokyo, Japan, vol. e84-a no.4, **(2001)**, pp. 1034-1041.

[12] S. Pabst, A. Koch and W. Straßer, "Fast and Scalable CPU/GPU Collision Detection for Rigid and Deformable Surfaces", In Computer Graphics Forum, Eurographics, vol. 29, no. 5, **(2010)**, pp. 1605-1612.

[13] T. Ning, L. J., Wang and 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, **(2012)**, pp. 389-397.

[14] 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, no. 1, **(2010)**, pp. 50-57.

[15] I. Grinberg and Y. Wiseman, "Scalable Parallel Collision Detection Simulation", In Proceedings of Signal and Image Processing, Honolulu, Hawaii, pp. 380-385, **(2007)**.

[16] R. Weller, "New Geometric Data Structures for Collision Detection and Haptics", Springer Science & Business Media, **(2013)**.

[17] I. Grinberg and Y. Wiseman, "Scalable Parallel Simulator for Vehicular Collision Detection", International Journal of Vehicle Systems Modelling and Testing, ISSN 1745-6436, vol. 8, no. 2, **(2013)**, pp. 119-144.

[18] U. Schwesinger, R. Siegwart and P. Furgale, "Fast collision detection through bounding volume hierarchies in workspace-time space for sampling-based motion planners", In IEEE International Conference on Proceedings of Robotics and Automation (ICRA), **(2015)**, pp. 63-68.

[19] L. Maarten and J. M. Phillips, "Shape fitting on point sets with probability distributions", In Algorithms-ESA 2009, Springer Berlin Heidelberg, **(2009)**, pp. 313-324.

[20] 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, **(2013)**, pp. 44-57.

[21] 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, **(2011)**, pp. 449-454.

[22] P. Weisberg and Y. Wiseman, "Efficient Memory Control for Avionics and Embedded Systems", International Journal of Embedded Systems, vol. 5(4), **(2013)**, pp. 225-238.

[23] Y. Wiseman, "Take a Picture of Your Tire!", Proceedings of IEEE Conference on Vehicular Electronics and Safety (IEEE ICVES-2010), Qingdao, ShanDong, China, **(2010)**, pp. 151-156.

[24] 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, **(2013)**, pp. 23-36.

[25] Y. Wiseman, "Camera That Takes Pictures of Aircraft and Ground Vehicle Tires Can Save Lives", Journal of Electronic Imaging, vol. 22, no. 4, **(2013)**, pp. 041104.

[26] D. H. Eberly, "3D Game Engine Design: A Practical Approach to Real-Time Computer Graphics", CRC Press, **(2006)**.

Author

Yair Wiseman, he is a Doctor and got a Summa Cum Laude M.Sc. and a PhD from Bar-Ilan University and completed two Post-Doc - one at the Hebrew University of Jerusalem and one in Georgia Institute of Technology.

Dr. Wiseman's research interests include Computational Transportation Science, Intelligent Transportation Systems, Process Scheduling, Hardware-Software Codesign, Memory Management, Computer Clusters, Data Compression, JPEG, Embedded Systems, Real-Time Systems and Operating Systems.

Dr. Wiseman is on the editorial board of several journals, a member of dozens of conference committees and a reviewer of many scholarly journals. Dr. Wiseman authored two books as well.

In addition, Dr. Wiseman has been teaching in many institutes including Bar-Ilan University, The Hebrew University of Jerusalem, Israel Aircraft Industry, Holon Institute of Technology and Jerusalem College of Technology.

Dr. Wiseman has been supervising many graduate students and an interesting point is that Albert Einstein is Dr. Wiseman's academic great-great-grandfather (*i.e.* the advisor of the advisor of the advisor of Dr. Wiseman's advisor).

Dr. Wiseman has collaborated with other partners and received research grants to run an active laboratory from inter alia Sun Microsystems, Intel, Polak Foundation and the Open University.

Dr. Wiseman is an international expert who has reviewed and evaluated several large projects of the European Union, Israel Science Foundation, MB Logic and more. Dr. Wiseman's papers have been published in many venues around the world.