Distance Computation between Convex Objects using Axis-Aligned Bounding-Box in Virtual Environment Application

Hamzah Asyrani Sulaiman[a], Abdullah Bade[b] and Mohd Harun Abdullah[c]

[a]*Universiti Teknikal Malaysia Melaka, Durian Tunggal, Melaka, Malaysia*
[a]*asyrani@utem.edu.my*
[b,c]*Universiti Malaysia Sabah, Kota Kinabalu, Sabah, Malaysia*
[b]*abade08@yahoo.com*, [c]*harun@ums.edu.my*

Abstract. Performing collision detection between convex objects in virtual environment simulation is one of vital problems in computer visualization research area. Given a set of two or more colliding objects, in order to determine the exact point of contact between object we need to undergo various high computation algorithm. In this paper, we describes our current work of determining the precise contact by measuring the distance between near colliding objects in order to maintain the accuracy and improve the speed of collision detection algorithm. Common method determine the distance by checking for vertices and edges point between objects in brute force condition. Compared to our method, by given set of objects in virtual environment world, we find the closest point between near colliding objects and bound the potential colliding area with an Axis-Aligned Bounding-Box. Then, we approximate the distance by measuring the distance of the box itself and hence recognize potential colliding area faster than the common method. Our method proven to most effective and efficient for narrow phase collision detection by removing unnecessary testing and reduced computational cost.

Keywords: Collision Detection, Axis-Aligned Bounding-Box, Virtual Environment, Distance Computation
PACS: 89.20.Ff, 07.05.Rm;

INTRODUCTION

Virtual environment application is consider as one of the key elements of bringing the uniqueness of computer graphics and visualization. Medical simulation, computer games development and many other virtual environment application has becoming essential requirement for programmers and system developers to well-designed their products before proceed with real or actual products [1-19]. It is a tool that help to investigate and perform pre-calculation of various probability with complex variables and parameters. Thus, it is very important to properly setup all the criteria of virtual environment application such as collision handling, lighting, texture complexity, and many others.

Determining the precise contact between two or more primitives is not an easy task as every time steps and movement of the objects must be calculated and computed. Hence, the collision detection algorithm must be properly designed in the simulation or computer games in order to set the realism of the surrounding environments. Some problem exists where the collision response can only manage to react once the collision has been collided before sending information that the objects has come into contact. The matter of this event is how long computer can receive input from the collided objects back to the collision detection mechanism. Important requirement such as time of contact, location of the contact and time lapse between reporting this event back is required in order to maintain the stability of the simulation. It also can be used and implemented in multiple applications and environments [20-22].

In this research, we concentrates on developing an efficient algorithm to compute the distance between two primitives. It means that the triangle of an object that is going to intersect with another triangle of other object. This research area is consider as part of narrow phase collision detection that consisting distance computation, point of contact, and depth penetration. In order to determine precisely the point of contact between two primitives, we need to speed up the process by minimizing the efforts of computing unnecessary points between primitives. Hence, by undergoing broad phase collision detection that used Bounding-Volume Hierarchies (BVH) to split up the complex object into smaller nodes containing group of triangles, we can check for potential candidates of primitives that is going to be intersected between leaf nodes (the lowest level of the BVH) that only contains single triangle and thus eliminating to check for non-intersecting nodes.

This paper consisting several sections that explain the procedure and experiment setup. Section II briefly explains the related works. Section III and IV shows our algorithm for distance computation and the experiment setup. Section V gives details analysis and discussion of our work. Section VI concluded our work.

RELATED WORK

Detecting object interference in 3D application is always challenging problem for researchers to overcome its limitation. Comparing to real world, performing fast and accurate collision detection is never an easy task as each object that contains triangles must be checked for interaction. For example, when a collision occurred between two geometric models that has 1000 triangles each of them, the corresponding system must check every triangle on one object between triangles in another object. Thus it is very expensive just to detect collision between only two objects that has only 1000 triangles. Any applications that have collision detection system must aware of this limitation and find a solution to overcome it. Apart from that, memory management in any 3D application is also must be considered. Furthermore, there are some other criteria and important issues need to be measured when designing a 3D application. For instance, in 3D games, animation and simulation, there are needs to maintain the application runs with suitable frame rates such as 30-60 fps [16, 23-31]. To achieve that goal, the applications should put high attention during construction phase through applying some appropriate techniques with some optimization. Instead of that, there is a need to balance between the realism and the complexity of the applications.

Discrete collision detection (DCD) works has been introduced since few decades ago when the speed of intersection between intersecting object need to be detected is vitally important. In 1988, Gilbert et al [19] created an efficient algorithm for computing the Euclidean distance between triangles using mathematical programming method where they used the position and orientation of the object and detect it along a continuous path. Later in 1990, Gilbert et al [32] improved it by using a simple extension of exploiting the capabilities of the polytope itself where it became more efficient than the previous algorithm. Later in 1994, Sean Quinlan [33] presented an efficient algorithm for DCD by using hierarchical representation of spheres. By enclosing the object with spheres for multiple levels, it helps to reduce the complexity by huge factors and also reduced computational cost.

By 1996, Hubbard introduced a sphere tree construction by using oct-tree based type. By using space-time bound technique, Hubbard technique allowed the collision detection technique working fast by approximation and graceful deprivation on the objects. In his dissertation, he concluded that the implementation of the algorithm works faster than the previous one.

Bounding-Volume Hierarchies of k-DOPs has been proposed by Klosowski et al [34] in 1998 by using a convex polytopes with some orientations of k value. By implemented this technique, their algorithm showed promising results that could benefits in complex static environment with collision detection algorithm. Other researchers also used various of BVs for the BVH such as boxtrees[35, 36] , Axis-Aligned Bounding-Box (AABBs) [12, 13, 24, 27, 35, 37-40], Oriented Bounding-Box (OBB) [4, 13, 26, 37, 39, 41-47], k-Dop [5] Oriented-Dops [42, 46, 48] and convex hulls [32, 33, 49, 50].

Distance Computation Algorithm for collision detection has been studied for the past three decades ago where M. Orlowski (1985) published a paper of "The Computation of the distance between polyhedra in 3-space", E.G. Gilbert, D.W. Johnson, and S.S. Keerthi (1988) published "A fast procedure for computing the distance between objects in three-dimensional space" and M.C. Lin (1991) published a popular paper of "A Fast Algorithm for Incremental Distance Calculation". Based on this paper, distance computation is mainly a method to determine the approximately high precision distance between pair of convex polyhedra. Distance computation algorithm is highly depends on the smallest step of object movement toward another objects as all computer simulation consists of coordination system. Thus, it has been used in another type of collision detection technique that focused on accuracy which is Continuous Collision Detection (CCD). Compared to Discrete Collision Detection (DCD), CCD provides a sequence of small, discrete steps that looks like a continuous movement.

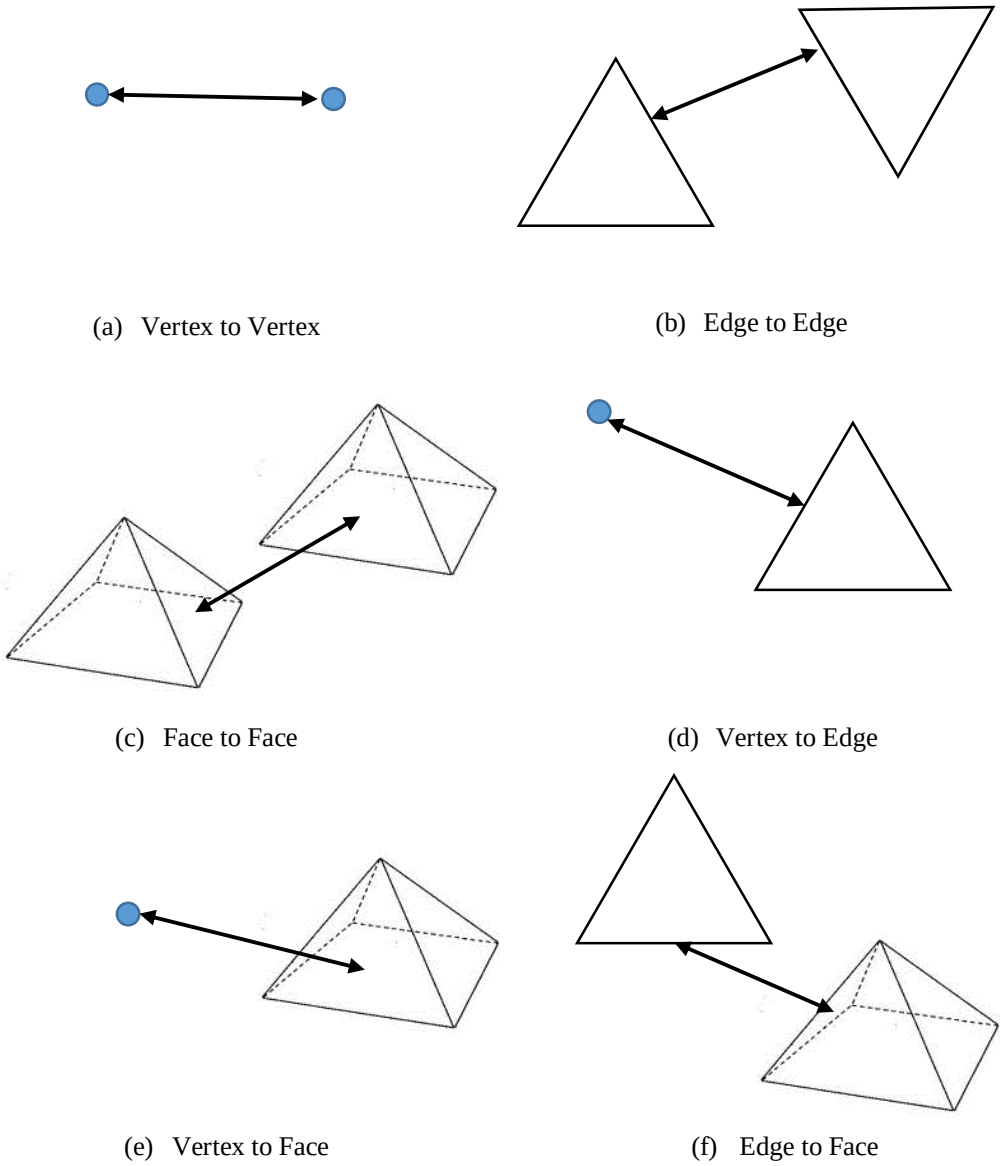

FIGURE 1. Example of six possible cases proposed by M.C. Lin in order to determine the closest features between two convex polyhedra

DISTANCE COMPUTATION ALGORITHM

In order perform collision detection simulation between objects, we created an empty space environment with two objects which are Stanford Library 3D model. Then, quadtree BVH is constructed down to the last piece of triangle, where each of triangle is bounded with an AABB using our stopping criteria. Figure 2 depicts our algorithm for computing distance between two near colliding triangle bound with AABB.

1. *Identify direction of object movement*
2. *Identify nearest midpoint of each object triangle that move toward the other.*
3. *Get minimum and maximum value of object movement axis*
    a. *Get first object minimum and maximum points*
    b. *Get second object minimum and maximum points*
4. *Construct Dynamic Inner AABB using 3(b) and 3(c)*
5. *Find midpoint and set as origin point*
6. *Calculate vertex to vertex and edge to vertex and edge to edge distance*
7. *Find shortest distance*

Figure 2: Dynamic origin Point (DyOP) Algorithm

EXPERIMENTS SETUP

In our experiments, we have conducted two tests that represented the speed of the distance computation and the accuracy of the distance by comparing with all the algorithms. We have loaded the environment with 10 types of triangles that is iterated with different types of triangle. From 10 types of triangle, we iterated for 90 times where each triangle will be tested against another shape of triangle.

Our experiments does not involve the same type of triangle in order to maintain the randomization of our algorithm with others. Moreover, it is a rare cases where each triangle in certain object will be tested against the same size. For example, if the object contains more than 1 million of triangle in random movement and angle, compute the distance with another object that also might have more than 1 million of triangle, thus, it will have a very minimum percentage that we will be having the same triangle size intersection. Hence in this experiments, we conducted 90 tests (where each triangle will be tested with the other nine thus resulting 90 tests) in order to perform analysis of DyOP, Lin Canny, and GJK algorithm.

From figure 3, our proposed algorithm of DyOP performance is better than the other two algorithms in term of algorithm speed for distance computation checks. All algorithms used the same fixed distance and the same tests. By improving the speed of distance computation algorithm, we can potentially increase the speed of collision detection testing especially the continuous collision detection (CCD) types where distance computation algorithm is mainly used in CCD application such as medical simulation and high precision simulation. Which means, more testing will be conducted when the speed is improved and maintaining the accuracy of the collision detection algorithm.

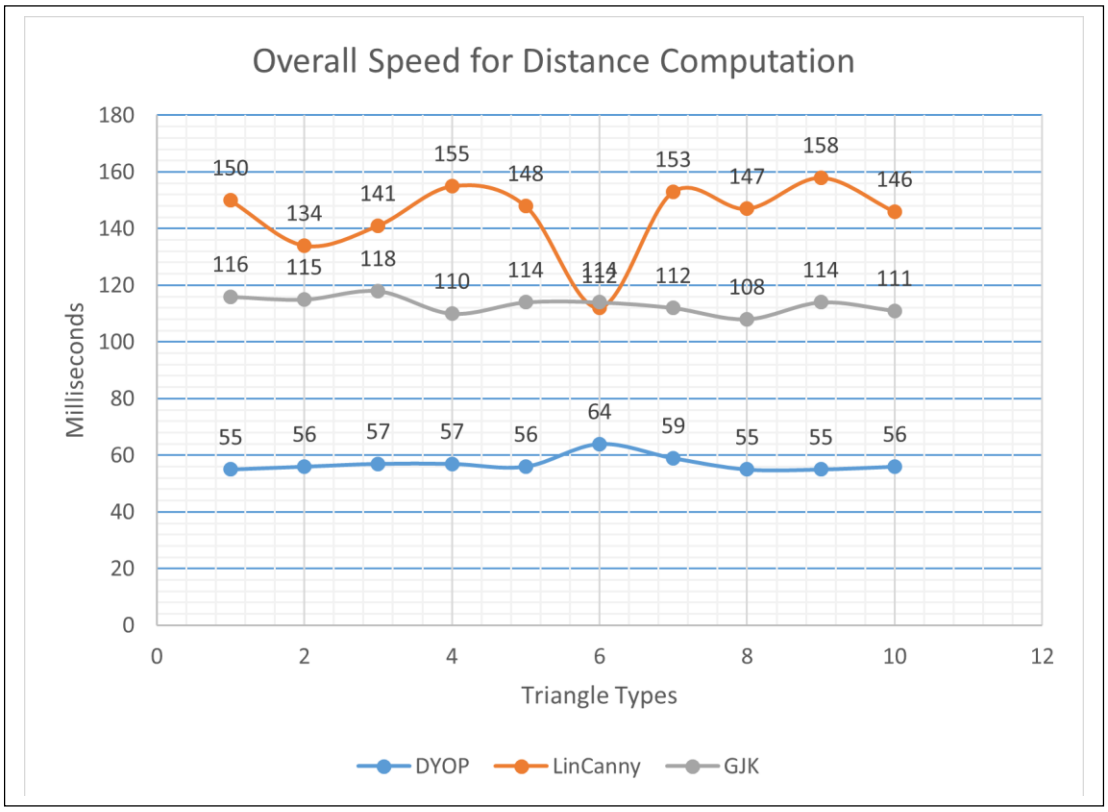

Figure 3: Overall Speed for Distance Computation

ANALYSIS AND DISCUSSION

Based on all experiments, we concluded that our proposed algorithm of DyOP is more superior to the other two algorithms of GJK and Lin-Canny. The speed of computing distance is increase between ranges of 150% to 180% for certain type of triangles. Although the implementation of GJK and Lin-Canny is following the implementation style of our code is not based on any other codes. But we followed strictly the concept of GJK and Lin-Canny in order to avoid any changes of the method. This is because there is numerous implementation style and code exists for both method but all just follow the same concept with their own style of implementation. However, in this research, all the concept and theoretical aspect of GJK and Lin-Canny is based on the original paper.

CONCLUSION AND FUTURE WORK

Our DyOP algorithm is implemented in 2D environment where it is the easiest part to see the contribution of this algorithm. It is the fundamental idea that we need to chase down to the bottom of complex model which is a triangle in order to visualize properly what is the idea of DyOP compared to any other methods. Moreover, it is to give other researcher about an overview about this new and novel algorithm at the fundamental level before implement it into any other applications. This algorithm also easy to implement in any other applications. However, our experiments is just based on distance computation only and not involving any collision response or penetration distance calculation.

ACKNOWLEDGMENTS

We would like to thanks for Universiti Teknikal Malaysia Melaka for providing PJP/2013/FKEKK/(15A)/S01199 for research grant. We also would to thank to Dr. Abdullah Bade and Prof. Mohd Harun Bin Abdullah of Universiti Malaysia Sabah for supervision and research expertise support.

REFERENCES

1. Qu, H. and W. Zhao. *Fast Collision Detection of Space-Time Correlation*. in *Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on*. 2012.
2. Hanwen, L., W. Yi, and M. Guanghui. *A CPU-GPU hybrid computing framework for real-time clothing animation*. in *Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference on*. 2011.
3. Yue, C., et al. *Phusis studio: A real-time physics engine for solid and fluid simulation*. in *Computational Problem-Solving (ICCP), 2011 International Conference on*. 2011.
4. Zhao, W. and L. Wang. *A fast collision detection algorithm suitable for complex virtual environment*. in *Transportation, Mechanical, and Electrical Engineering (TMEE), 2011 International Conference on*. 2011.
5. Zhang, P. and g.-l. Du. *A fast continuous collision detection algorithm based on K_DOPs*. in *Electronics, Communications and Control (ICECC), 2011 International Conference on*. 2011.
6. Sulaiman, H.A., A. Bade, and N.M. Suaib. *Balanced hierarchical construction in collision detection for rigid bodies*. in *Science and Social Research (CSSR), 2010 International Conference on*. 2010.
7. Sulaiman, H.A., A. Bade, and N.M. Suaib. *Bounding-Volume Hierarchies Technique for Detecting Object Interference in Urban Environment Simulation*. in *Second International Conference on Environmental and Computer Science, 2009. ICECS '09*. 2009.
8. Tang, M., et al., *Interactive continuous collision detection between deformable models using connectivity-based culling*, in *Proceedings of the 2008 ACM symposium on Solid and physical modeling*. 2008, ACM: Stony Brook, New York. p. 25 - 36.
9. Sulaiman, H.A., et al., *Bounding Volume And How To Create Them*, in *Collision Detection for Real Time Computer Graphics : Series of Techniques*, A. Bade and N. Suaib, Editors. 2008, Penerbit UTM: Johor Bahru, Johor. p. 19 - 30.
10. Sulaiman, H.A., N.M. Suaib, and D. Daman, *Bounding Volume Hierarchies For Collision Detection*, in *Collision Detection for Real Time Computer Graphics : Series of Techniques*, A. Bade and N. Suaib, Editors. 2008, Penerbit UTM: Johor Bahru, Johor. p. 65 - 80.
11. Tuft, D.O., *A System For Collision Detection Between Deformable Models Built On Axis Aligned Bounding Boxes And Gpu Based Culling*, in *Department of Computer Science*. 2007, Brigham Young University: Brigham. p. 73.
12. Weller, R.e., J. Klein, and G. Zachmann. *A Model for the Expected Running Time of Collision Detection using AABB Trees*. in *Eurographics Symposium on Virtual Environments (EGVE)*. 2006. Lisbon, Portugal.
13. Zhiwen, Y. and W. Hau-San. *GPCD: Grid-based Predictive Collision Detection for Large-scale Environments in Computer Games*. in *Multimedia and Expo, 2006 IEEE International Conference on*. 2006.
14. Benitez, A., M.d.C. Ramirez, and D. Vallejo, *Collision Detection Using Sphere-Tree Construction*, in *Proceedings of the 15th International Conference on Electronics, Communications and Computers*. 2005, IEEE Computer Society. p. 286 - 291.
15. Sanna, A. and M. Milani. *CDFast: an Algorithm Combining Different Bounding Volume Strategies for Real Time Collision Detection*. in *SCI*. 2004.
16. Redon, S., *Fast continuous collision detection and handling for desktop virtual prototyping*. Virtual Reality, 2004. **8**(1): p. 63-70.
17. Redon, S., et al., *Fast continuous collision detection for articulated models*, in *Proceedings of the ninth ACM symposium on Solid modeling and applications*. 2004, Eurographics Association: Genoa, Italy.
18. Zachmann, G., *Virtual Reality in Assembly Simulation - Collision Detection, Simulation Algorithms, and Interaction Techniques*, in *Department of Computer Science*. 2000, Darmstadt University of Technology: Germany.
19. Gilbert, E.G., D.W. Johnson, and S.S. Keerthi, *A fast procedure for computing the distance between complex objects in three-dimensional space*. Robotics and Automation, IEEE Journal of, 1988. **4**(2): p. 193-203.

20. Zakaria, Z., et al., *Current developments of microwave filters for wideband applications.* World Applied Sciences Journal, 2013. **21**(SPECIAL ISSUE2): p. 31-40.
21. Othman, M.A., M.Z.A.A. Aziz, and M. Sinnappa. *The exposure of Radio Frequency (RF) radiation on pharmaceuticals medicine for RF application*. 2012.
22. Ismail, M.M., et al. *Firefly algorithm for path optimization in PCB holes drilling process*. 2012.
23. Gao, Y., et al., *View-Dependent Multiscale Fluid Simulation.* Visualization and Computer Graphics, IEEE Transactions on, 2012. **PP**(99): p. 1-1.
24. Gong, J., J. An, and L. Cui. *Research and Application for Collision Detection Algorithm in Virtools*. in *Business Computing and Global Informatization (BCGIN), 2011 International Conference on*. 2011.
25. Kaiqiang, G. and X. Jiewu. *An Improved Algorithm of Collision Detection in 2D Grapple Games*. in *Intelligent Information Technology and Security Informatics (IITSI), 2010 Third International Symposium on*. 2010.
26. Lu, C. and Q. Guofeng. *Optimization of the collision detection technology in 3D skeleton animation*. in *Computer Application and System Modeling (ICCASM), 2010 International Conference on*. 2010.
27. Yi-Si, X., X.P. Liu, and X. Shao-Ping. *Efficient collision detection based on AABB trees and sort algorithm*. in *Control and Automation (ICCA), 2010 8th IEEE International Conference on*. 2010.
28. Sulaiman, H.A., et al. *Bounding-volume hierarchies for detecting interference in urban simulation*. in *Proc. of Computer Games & Allied Technology in Animation, Multimedia, IPTV & Entertainment*. 2008. Singapore.
29. Enhua, W. *Physically Based Simulation of Fluid Mixtures*. in *Computer-Aided Design and Computer Graphics, 2007 10th IEEE International Conference on*. 2007.
30. Stylianou, S., M.M. Fyrillas, and Y. Chrysanthou, *Scalable pedestrian simulation for virtual cities*, in *Proceedings of the ACM symposium on Virtual reality software and technology*. 2004, ACM: Hong Kong. p. 65 - 72.
31. Arnaldo, C. and R. Paola, *Is urban gaming simulation useful?* Simul. Gaming, 2001. **32**(4): p. 507-521.
32. Gilbert, E.G. and C.P. Foo, *Computing the distance between general convex objects in three-dimensional space.* Robotics and Automation, IEEE Transactions on, 1990. **6**(1): p. 53-61.
33. Quinlan, S. *Efficient distance computation between non-convex objects.* in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*. 1994.
34. Klosowski, J.T., et al., *Efficient Collision Detection Using Bounding Volume Hierarchies of k-DOPs.* IEEE Transactions on Visualization and Computer Graphics, 1998. **4**(1): p. 21-36.
35. Okada, K., M. Inaba, and H. Inoue. *Real-time and Precise Self Collision Detection System for Humanoid Robots.* in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. 2005.
36. Wang, X.-p., et al. *Solution to collision detection based on bounding box in Vega Prime*. in *Future Information Technology and Management Engineering (FITME), 2010 International Conference on*. 2010.
37. Feixiong, L., et al. *Real-time collision detection and response in virtual global terrain environments*. in *Computer-Aided Industrial Design & Conceptual Design, 2009. CAID & CD 2009. IEEE 10th International Conference on*. 2009.
38. Hanwen, L. and W. Yi. *Coherent hierarchical collision detection for clothing animation*. in *Haptic Audio Visual Environments and Games (HAVE), 2011 IEEE International Workshop on*. 2011.
39. Tu, C. and L. Yu. *Research on Collision Detection Algorithm Based on AABB-OBB Bounding Volume*. in *First International Workshop on Education Technology and Computer Science, 2009. ETCS '09.* . 2009.
40. Zhang, X. and Y.J. Kim, *Interactive Collision Detection for Deformable Models Using Streaming AABBs.* IEEE Transactions on Visualization and Computer Graphics, 2007. **13**(2): p. 318-329.
41. Chun-yan, Y., et al. *A new horizontal collision detection scheme for avatar with avatar in collaborative virtual environment*. in *Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on*. 2005.
42. Yanchun, S. and S. Xingyi. *Research and improvement of collision detection based on oriented bounding box in physics engine*. in *Communication Software and Networks (ICCSN), 2011 IEEE 3rd International Conference on*. 2011.
43. Chang, J.-W., W. Wang, and M.-S. Kim, *Efficient collision detection using a dual OBB-sphere bounding volume hierarchy.* Computer-Aided Design, 2010. **42**(1): p. 50-57.
44. Zhou, X. *Research of collision detection based on OBB in skinned mesh*. in *Computer Application and System Modeling (ICCASM), 2010 International Conference on*. 2010.

45. Shen, X.-l. and J.-s. Zhang. *Research of collision detection algorithm based on particle swarm optimization*. in *Computer Design and Applications (ICCDA), 2010 International Conference on*. 2010.
46. Chang, J.-W., W. Wang, and M.-S. Kim, *Efficient collision detection using a dual OBB-sphere bounding volume hierarchy*. Computer-Aided Design, 2009. **In Press, Corrected Proof**.
47. Gottschalk, S., M.C. Lin, and D. Manocha, *OBBTree: a hierarchical structure for rapid interference detection*, in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 1996, ACM.
48. Bade, A., et al., *Oriented convex polyhedra for collision detection in 3D computer animation*, in *Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia*. 2006, ACM: Kuala Lumpur, Malaysia.
49. Zhang, X., M. Lee, and Y.J. Kim, *Interactive continuous collision detection for non-convex polyhedra.* Vis. Comput., 2006. **22**(9): p. 749-760.
50. Cameron, S., *A comparison of two fast algorithms for computing the distance between convex polyhedra.* Robotics and Automation, IEEE Transactions on, 1997. **13**(6): p. 915-920.