Distance Computation using Axis Aligned Bounding Box (AABB) Parallel Distribution of Dynamic Origin Point

Hamzah Asyrani Sulaiman[1], Mohd Azlishah Othman[2], Mohd Muzafar Ismail[3], Maizatul Alice Meor Said[4], Ridza Azri Ramlee[5], Mohamad Harris Misran[6]
CeTRi
Universiti Teknikal Malaysia Melaka
Durian Tunggal, Melaka, Malaysia
[1]asyrani@computer.org, [2]azlishah@utem.edu.my, [3]muzafar@utem.edu.my, [4]alice@utem.edu.my, [5]ridza@utem.edu.my, [6]harris@utem.edu.my

Abdullah Bade[7], Mohd Harun Abdullah[8]
School of Science and Technology
Universiti Malaysia Sabah
Kota Kinabalu, Sabah
[7]abade08@yahoo.com, [8]harun@ums.edu.my

***Abstract*— Performing accurate and precise collision detection method be-tween objects in virtual environment application such as computer games and medical simulation is important in computer graphics research and development. Given pair of objects that near colliding, numerous mechanic has been developed by researchers in order to minimize computation time and increase accuracy of the detection. However, most of these techniques required a lot of computational cost, extra processing power and complex algebraic equations just to solve distance between near colliding objects. In this paper, we described an alternate technique, which is a theoretical framework of novel technique in order to find the optimum closest distance between two or more convex polyhedral in virtual environment application. Given pair of near colliding objects, we proposed an easy to implement mechanism using dynamic origin point by creating inner and middle Axis Aligned Bounding-Box just to find closest distance between objects. We believed that the technique is suitable to be used in any game engine tools for com-puter games and medical simulation.**

Keywords— collision detection, virtual environment, distance computation, computer graphics

I. Introduction

The development of virtual environment application has become one of the recent trends especially in computer graphics research area. In order to develop a well-structured 3D virtual environment application, numerous aspects has to be considered before the real event take place, as many important factors has to be checked. Criteria such as collision detection, lighting, texturing, speed, objects complexity, and some other techniques are the main consideration for virtual environment application. For medical simulation, an accurate collision technique along with the high-definition texturing technique and superb lighting technique are the main important factors. Meanwhile, computer games development requires sophisticated physic system with fast collision detection depending on what type of games that the developers are trying to develop. Hence, this is why the 3D computer graphics is one of the 21st century's research areas that could bring profits to most of the top computer graphics companies.

Determining the precise contact between two or more polyhedral is not an easy task as every time steps and movement of the objects must be calculated and computed. Hence, the collision detection system must be properly installed in the simulation or computer games in order to maintain the realism of the surrounding environments. However, the 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 input 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.

In this research, we have proposed a theoretical framework on how to precisely know the exact distance between two convex polyhedral compared to other methods. Instead of performing complex algebraic solution that has been used in various techniques to determine the distance computation algorithm in determining collision, we have proposed a technique that use the efficiency of Axis-Aligned Bounding-Box (AABB) that create boundary surrounding the last piece of triangle in Bounding-Volume Hierarchies (BVH). This technique is called Dynamic Object Point for Distance Computation (DOPDC) technique. It explicitly used the inner and outer AABB capabilities that have been created using dynamic parallel lines. Section II will describe about previous works on collision detection and distance computation algorithms. Section III explained in detailed regarding Bounding-Volume Hierarchies and the theoretical framework and the pseudo code of the algorithm along with the analysis. Section IV conclude our framework.

978-1-4673-5149-2/13/$31.00 ©2013 IEEE

II. RESEARCH BACKGROUND

Collision detection research studies have been performed well by many top researchers in various fields such as computational geometry, robotics, computer games and animation and computer graphics [1-19]. Some of the main references for other researcher in term of survey papers are [17, 20-27]. Hence in this section, we described few selected technique that are related to the work of collision detection especially in discrete and continuous collision detection.

A. Discrete Collision Detection

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 [28] 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 [29] 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 [30] 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[3, 31] , Axis-Aligned Bounding-Box (AABBs) [3, 16, 32-38], Oriented Bounding-Box (OBB) [5, 12, 32, 33, 37, 39-45], k-Dop [46] Oriented-Dops [12, 44, 47] and convex hulls [28, 29, 48, 49].

B. Continuous Collision Detection

Continuous Collision Detection (CCD) works differently by Discrete Collision Detection (DCD) algorithm. Instead of using time step to recognize the potential colliding information, CCD used the object movement towards other object in smallest approximation value. Hence, by knowing the distance between two or more object using either algebraic equation or other technique, we can approximate precise contact between colliding object. CCD is the most accurate solution for real-time simulation such as medical simulation and high precision measurement simulation. However, the computational cost for CCD is far more expensive than the DCD methods. Apparently, most of these approaches seem unable to perform very fast CCD queries compared to DCD queries on general polygonal models. Only few of those can handle polygon-soup models such as [18, 50]. In Redon et al.[50], they extend capabilities of static OBB tree algorithm used in [45] by modifying the separating axis theorem. They create a continuous version of separating axis theorem and demonstrate real-time performance on polygonal models. However, it creates another problem as it becomes excessively conservative when it involves large rotations between two objects. It occurs when two objects has extreme collision that almost huge part of the second object intersects with first object. Compared to Zhang et al.'s algorithm [48], it much slower for polyhedral models. They already implemented the algorithm in articulated models [51], but it still use the same fundamental which is polyhedron. Redon et al. [52] illustrate an extension of [53] to articulated models and their algorithm also had similar problem. Schwarzer et al.'s algorithm [54] is based on a conservative condition to guarantee a collision-free motion linking with two configurations, but the condition may become excessively conservative when an object slides over another object.

III. THEORITICAL FRAMEWORK

In this paper we will describes our theoretical framework of our propose technique that use the capabilities of Axis-Aligned Bounding-Box to calculate the distance between two objects down to the distance between two triangles.

A. Region Creation

At first, from the Bounding-Volume Hierarchies (BVH) that enclosed the triangle down to the last piece of triangle, we will use the AABB for each of the triangle that approximately close to each other. By maintaining one of the axes that has the shortest distance between two AABBs, we can create an invisible AABB just to enclose both of BV into one (combined). By doing this, we can get a new region that has two triangle that might close together to create a contact or determine the contact between them. Figure 1 explain how one of the axes can approximately know which triangles that most likely can come into contact and then determine the distance.

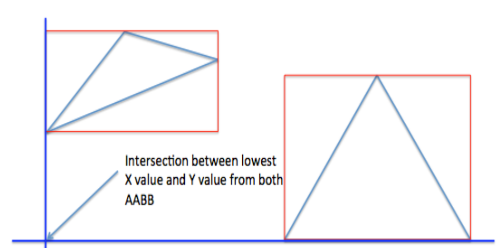

Figure 1 From the figure, both triangles can be recognized their potential axes in order to collide

B. Parallel Line Elimination using Outer AABB techniques

Parallel Line Elimination (PLE) stands for elimination of the axes lines that we do not require for distance computation between both triangles. In this case, we need to search for any lines that we do not need for intersection and distance computation testing. Figure 2 explains the situation.

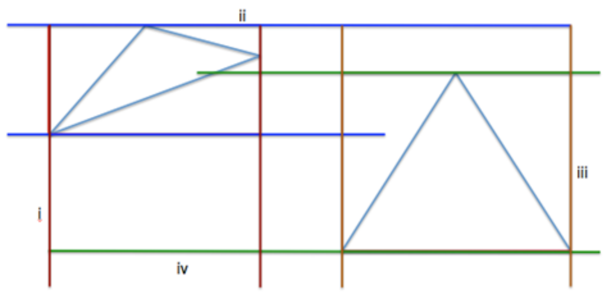

Figure 2 Parallel Line Elimination process to remove any lines that we do not require for nearest intersection contact.

From the Figure 2, we found that using the maximum and minimum point that created AABB, we can do the elimination. Given A_{xmin} and A_{xmax} as minimum and maximum points of first triangle (the leftmost), and B_{xmin} and B_{xmax} for the second triangle (the rightmost). First we will perform comparison between A_{xmin} and B_{xmin} in order to determine which one will be having the greater or lower value. We perform elimination or remove the lower value, as it is not require for distance computation algorithm. This only applies for X-axis where the nearest distance most likely to lies between A_{xmax} and B_{xmin}. Thus in this case A_{xmin} is lower that B_{xmin} and we will select A_{xmin} as candidate to be removed from our distance computation calculation. The same thing applies for maximum points comparison between both triangles in X-axis where between A_{xmax} and B_{xmax}, only B_{xmax} will be removed from our calculation as the B_{xmax} is largest value of all X-axis. In short, between A_{xmin} , A_{xmax} , B_{xmax}, and B_{xmin}, remove any points that has highest and lowest X-axis values thus leaving only the X-axis point that most likely the nearest of X-axis that can come into contact.

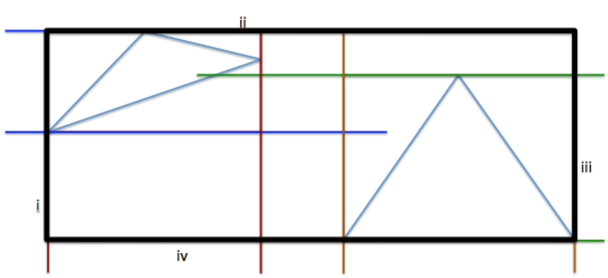

Figure 3 Outer AABB for removing those lines that are not required for Distance Computation Calculation

Outer AABB is one of the techniques that can be used to eliminate those lines that we do not need in order to calculate the distance between those two triangles. By finding the maximum and minimum points of all triangles (that we used to create AABB for each triangle), we can create a new Outer AABB to find any points that belong to Outer AABB creation. In previous paragraph we have explained how to find the lowest and highest value for each axis. However, we can perform faster elimination by performing Outer AABB technique. Figure 3 shows Outer AABB technique.

It works be enclosing the triangle that might come into contact (nearest) with the big AABB. Then, once we have found points that created Outer AABB (big AABB), we can determine which point that most likely to create them. In this case, A_{ymax}, and B_{ymin} are belongs to Outer AABB creation and thus we removed them from our distance computation calculation. Figure 4 shows the final configuration for us to compute the distance.

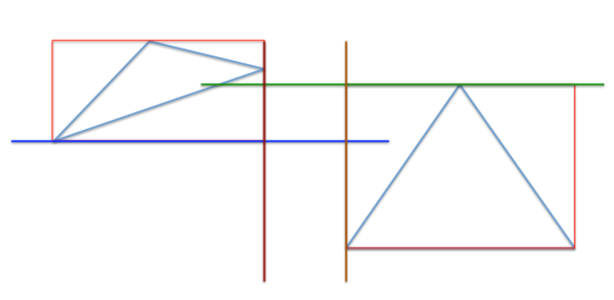

Figure 4 Final configuration of Distance Computation calculation where points/lines that close to only the nearest intersection of area that might come into contact is required

C. Midpoint of AABB

From a closer look at the 'empty box' that lies between the two triangles in AABB, we now can find the midpoint of the AABB as one of the main component for distance computation calculation. We will then transforming the old origin point located at the coordinates (0,0,0) into the new one (let say $X_{neworigin}$, $Y_{neworigin}$, and $Z_{neworigin}$). Figure 5 explains the procedure to determine a new origin point for distance computation algorithm that we develop.

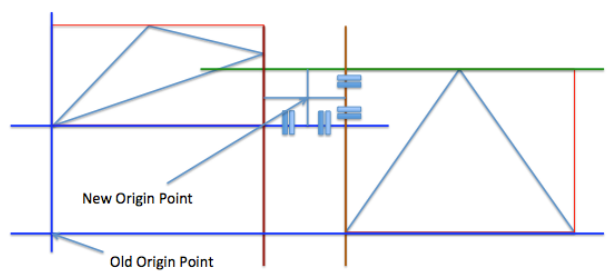

Figure 5 New Origin Point

From Figure 5, a New Origin Point (NOP) can easily be calculated by using the formula below:

$$\frac{B_{xmin} - A_{xmax}}{2} = NOP_{x-mid}$$

the same process will happens to the y coordinate by changing it to:

$$\frac{B_{ymax} - A_{ymin}}{2} = NOP_{y-mid}$$

Once NOP has been calculated, we will save it into the memory for distance computation calculation after this.

D. Figures and Tables Middle Box of AABB for Intersection Points

The Middle Box of AABB is an “empty box” that temporarily created to find the midpoint of rectangle in order to initialize NOP. Once the NOP has been saved into memory, the process of distance computation calculation continues with the process of finding the nearest side of triangle that may come into contact. Figure 6 shows the corresponding behavior that need to be check in order to find the nearest side of triangle.

$$y_{edgeleft} = \frac{N_{xmax} - N_{ymin}}{M_{xmax} - M_{ymin}} x + C_A$$

$$y_{rightleft} = \frac{Q_{ymax} - Q_{xmin}}{P_{ymax} - P_{xmin}} x + C_B$$

where C_A *and* C_B is the interception of y axis that need to be calculated first.

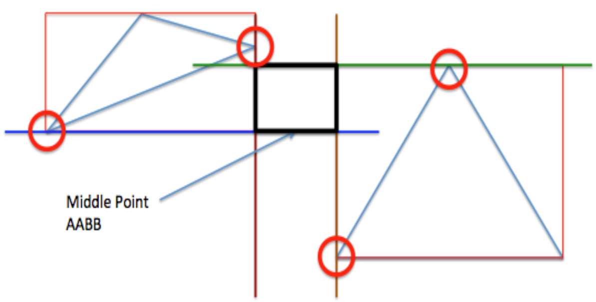

Figure 6 Middle Box of AABB located just between two triangles and contains Middle Point AABB or NOP

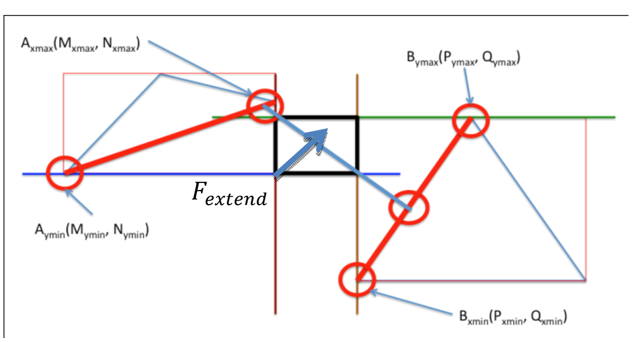

Figure 7 The mechanism to detect potential edges by using points that created AABB of the triangles

From the Figure 6, the process of finding the right edge of intersection area started by using all points that created the Middle Box of AABB. By utilizing the A_{xmax}, A_{ymin}, B_{xmin}, and B_{ymax}, we can easily create an edge and thus it can be declared as the nearest edge that might come into contact. Line equations for both edges must be calculated in order to obtain points or triangle that intersected with a line that is created from the NOP to both edges. From Figure 7, we can obtains two line equations

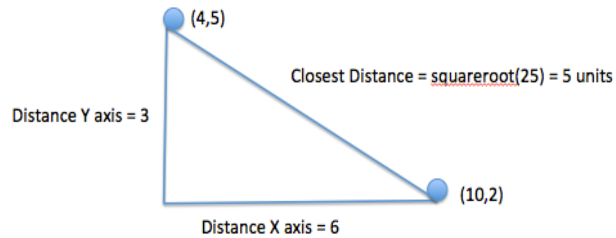

Figure 8 Trigonometric rules of finding the distance between two points

Conclusion and Future Work

We have presented our current research progress on development of a novel algorithm of Distance Computation using Axis Aligned Bounding Box (AABB) Parallel Distribution of Dynamic Origin Point. Currently, the experimental process in being carried out using C++ programming language in Visual Studio C++ 2010. Stanford 3D model has been used in our experiments in order to maintain the standard 3D object that had been used for collision detection testing.

Acknowledgment

We would like to thank for Centre for Telecommunication Research and Innovation (CeTRi) for research feedback and Dr. Abdullah Bade and Prof. Datuk Dr. Mohd Harun Abdullah from Universiti Malaysia Sabah for PhD supervision.

References

[1] B. Chen, X. Ye, L. An, and Y. Wang, "Detection of Collision and Self-Collision Using QPSO for Deformable Models," in *Intelligent System Design and Engineering Application (ISDEA), 2012 Second International Conference on*, 2012, pp. 1028-1031.

[2] H. Qu and W. Zhao, "Fast Collision Detection of Space-Time Correlation," in *Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on*, 2012, pp. 567-571.

[3] O. Arcila, S. Dinas, and J. M. Banon, "Collision detection model based on Bounding and containing Boxes," in *Informatica (CLEI), 2012 XXXVIII Conferencia Latinoamericana En*, 2012, pp. 1-10.

[4] W. Zhao and X. Guo, "The Collision Detection Algorithm in Virtual Reality," in *Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on*, 2012, pp. 538-541.

[5] Z. Wei and Z. Ying, "Survey of Collision Detection in Roaming of Virtual Environment," in *Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on*, 2012, pp. 552-556.

[6] O. Fan and Z. Tie, "Octree-based Spherical hierarchical model for Collision detection," in *Intelligent Control and Automation (WCICA), 2012 10th World Congress on*, 2012, pp. 3870-3875.

[7] K. W. Ng, Y. W. Yeap, Y. H. Tan, and K. I. Ghauth, "Collision detection optimization on mobile device for shoot'em up game," in *Computer & Information Science (ICCIS), 2012 International Conference on*, 2012, pp. 464-468.

[8] J. Byung-jin, C. Hyouk Ryeol, K. Ja Choon, and M. Hyungpil, "Collision detection using band designed Disturbance Observer," in *Automation Science and Engineering (CASE), 2012 IEEE International Conference on*, 2012, pp. 1080-1085.

[9] L. Jie and A. Ephremides, "A New Approach to Random Access: Reliable Communication and Reliable Collision Detection," *Information Theory, IEEE Transactions on,* vol. 58, pp. 989-1002, 2012.

[10] R. E. Sofronia, G. G. Savii, and A. Davidescu, "Real-time collision detection for long thin medical instruments in virtual reality-based simulators," in *Optimization of Electrical and Electronic Equipment (OPTIM), 2012 13th International Conference on*, 2012, pp. 1107-1112.

[11] Z. Wei and L. Lei, "Improved K-DOPs collision detection algorithms based on genetic algorithms," in *Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference on*, 2011, pp. 338-341.

[12] S. Yanchun 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, pp. 73-76.

[13] H. A. Sulaiman, 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, pp. 1132-1136.

[14] H. A. Sulaiman, A. Bade, D. Daman, and N. M. Suaib, "Collision Detection using Bounding-Volume Hierarchies in Urban Simulation," presented at the The 5th Postgraduate Annual Research Seminar, Faculty of Computer Science & Information System, UTM, 2009.

[15] N. M. Suaib, A. Bade, and D. Mohamad, "Collision Detection Using Bounding-Volume for avatars in Virtual Environment applications," in *The 4th International Conference on Information & Communication Technology and Systems*, Institut Teknologi Sepuluh Nopember (ITS), Surabaya, Indonesia, 2008, pp. 486 - 491.

[16] R. e. Weller, 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)*, Lisbon, Portugal, 2006.

[17] C. Ericson, *Real-Time Collision Detection (The Morgan Kaufmann Series in Interactive 3-D Technology) (The Morgan Kaufmann Series in Interactive 3D Technology)*: Morgan Kaufmann Publishers Inc., 2004.

[18] S. Redon, A. Kheddar, and S. Coquillart, "CONTACT: arbitrary in-between motions for collision detection," in *10th IEEE International Workshop on Robot and Human Interactive Communication, 2001*, 2001, pp. 106-111.

[19] E. G. Gilbert, 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,* vol. 4, pp. 193-203, 1988.

[20] H. A. Sulaiman, 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, pp. 436-440.

[21] M. Tang, S. Curtis, S.-E. Yoon, and D. Manocha, "Interactive continuous collision detection between deformable models using connectivity-based culling," presented at the Proceedings of the 2008 ACM symposium on Solid and physical modeling, Stony Brook, New York, 2008.

[22] S. H. Kockara, T.; Iqbal, K.; Bayrak, C.; Rowe, Richard;, "Collision Detection - A Survey," presented at the IEEE International Conference on Systems, Man and Cybernetics, 2007. ISIC., 2007.

[23] G. V. D. Bergen, *Collision Detection in Interactive 3D Environments*. United States of America: Elsevier, Inc., 2004.

[24] M. C. Lin and D. Manocha, "Collision and Proximity Queries," in *In Handbook of Discrete and Computational Geometry, 2nd Ed*. vol. 35, ed Boca Raton, FL: CRC Press LLC, 2004, pp. 787-807.

[25] S. Redon, Y. J. Kim, M. C. Lin, and D. Manocha, "Fast continuous collision detection for articulated models," presented at the Proceedings of the ninth ACM symposium on Solid modeling and applications, Genoa, Italy, 2004.

[26] S. A. Gottschalk, "Collision queries using oriented bounding boxes," The University of North Carolina at Chapel Hill, 2000.

[27] G. Zachmann, "Virtual Reality in Assembly Simulation - Collision Detection, Simulation Algorithms, and Interaction Techniques," Department of Computer Science, Darmstadt University of Technology, Germany, 2000.

[28] E. G. Gilbert and C. P. Foo, "Computing the distance between general convex objects in three-dimensional space," *Robotics and Automation, IEEE Transactions on,* vol. 6, pp. 53-61, 1990.

[29] S. Quinlan, "Efficient distance computation between non-convex objects," in *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, 1994, pp. 3324 - 3329.

[30] J. T. Klosowski, M. Held, J. S. B. Mitchell, H. Sowizral, and K. Zikan, "Efficient Collision Detection Using Bounding Volume Hierarchies of k-DOPs," *IEEE Transactions on Visualization and Computer Graphics,* vol. 4, pp. 21-36, 1998.

[31] X.-p. Wang, X.-c. Dong, H.-b. Wang, H.-c. Zheng, and S.-f. Gu, "Solution to collision detection based on bounding box in Vega Prime," in *Future Information Technology and Management Engineering (FITME), 2010 International Conference on*, 2010, pp. 113-116.

[32] Z. Wei and S. Jing, "Collision Detection Research for Deformable Objects," in *Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on*, 2012, pp. 557-561.

[33] H. Rui, "Optimizing collision detection in 3D games with model attribute and Bounding Boxes," in *Electrical & Electronics Engineering (EEESYM), 2012 IEEE Symposium on*, 2012, pp. 589-591.

[34] L. Hanwen and W. Yi, "Coherent hierarchical collision detection for clothing animation," in *Haptic Audio Visual Environments and Games (HAVE), 2011 IEEE International Workshop on*, 2011, pp. 129-134.

[35] J. Gong, 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, pp. 457-460.

[36] X. Yi-Si, 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, pp. 328-332.

[37] C. Tu 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, pp. 331-333.

[38] X. Zhang and Y. J. Kim, "Interactive Collision Detection for Deformable Models Using Streaming AABBs," *IEEE Transactions on Visualization and Computer Graphics,* vol. 13, pp. 318-329, 2007.

[39] W. Zhao 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, pp. 502-505.

[40] 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, pp. 50-57, 2010.

[41] C. Lu 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, pp. V10-539-V10-543.

[42] X. Zhou, "Research of collision detection based on OBB in skinned mesh," in *Computer Application and System Modeling (ICCASM), 2010 International Conference on*, 2010, pp. V6-643-V6-645.

[43] X.-l. Shen 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, pp. V1-60-V1-63.

[44] J.-W. Chang, W. Wang, and M.-S. Kim, "Efficient collision detection using a dual OBB-sphere bounding volume hierarchy," *Computer-Aided Design,* vol. In Press, Corrected Proof, 2009.

[45] S. Gottschalk, M. C. Lin, and D. Manocha, "OBBTree: a hierarchical structure for rapid interference detection," presented at the Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, 1996.

[46] P. Zhang 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, pp. 617-621.

[47] A. Bade, N. Suaib, M. Z. A, and T. S. T. M, "Oriented convex polyhedra for collision detection in 3D computer animation," presented at the Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia, Kuala Lumpur, Malaysia, 2006.

[48] X. Zhang, M. Lee, and Y. J. Kim, "Interactive continuous collision detection for non-convex polyhedra," *Vis. Comput.,* vol. 22, pp. 749-760, 2006.

[49] S. Cameron, "A comparison of two fast algorithms for computing the distance between convex polyhedra," *Robotics and Automation, IEEE Transactions on,* vol. 13, pp. 915-920, 1997.

[50] S. Redon, A. Kheddar, and S. Coquillart, "Fast Continuous Collision Detection between Rigid Bodies," *Computer Graphics Forum,* vol. 21, pp. 279-287, 2002.

[51] A. Wong, "Adaptive bilateral filtering of image signals using local phase characteristics," *Signal Processing,* vol. 88, pp. 1615-1619, 2008.

[52] J. Zhou, H. Zhu, H. Shu, and L. Luo, "A generalized diffusion based inter-iteration nonlinear bilateral filtering scheme for PET image reconstruction," *Computerized Medical Imaging and Graphics,* vol. 31, pp. 447-457, 2007.

[53] N. Bhoi and S. Meher, "Circular spatial filtering under high-noise-variance conditions," *Computers & Graphics,* vol. 32, pp. 568-580, 2008.

[54] J. Luo and M. Boutell, "Natural scene classification using overcomplete ICA," *Pattern Recognition,* vol. 38, pp. 1507-1519, 2005.