The Construction of Balanced Bounding-Volume Hierarchies using Spatial Object Median Splitting Method for Collision Detection

Hamzah Asyrani Sulaiman[1], and Abdullah Bade[2]

[1]Universiti Teknikal Malaysia Melaka,
76100 Durian Tunggal, Melaka, Malaysia,
[2] Universiti Malaysia Sabah,
88400 Kota Kinabalu, Sabah, Malaysia,
[1]asyrani@utem.edu.my , [2] abade08@yahoo.com

ABSTRACT

Finding two or more contact points between rigid bodies simulation is always a fundamental task in virtual environment. Furthermore, the contact point needs to be accurately reported as soon as possible within 30-60 frames per second (fps) between moving polyhedral. This article introduced an efficient splitting method that is able to divide the bounding-volume of Axis Aligned Bounding-Box (AABB) hierarchies into a balanced tree. The construction of well-balanced tree will helps to improve the speed of the intersection between rigid bodies' objects.

KEYWORDS

Collision Detection, Bounding-Volume Hierarchies, Algorithm

1 INTRODUCTION

Creating real-time virtual environment consisting various 3D objects such as rigid bodies, deformable bodies and fluid simulation definitely a huge task for designers and researchers. Numerous strategies in developing such environment need to be considering before the simulation is suitable to be running in the respective platform i.e. targeting computer. In the simulated virtual environment, a rigid object isconsidered as a static object that cannot be deformed no matter what artificial force is been done. For example, buildings cannot be crumbled into rubble when anything happens. Meanwhile, a deformable object is something that can be altered in their shape according to the time and for that has been applied to them. Fluid simulation consists of any fluid mechanism that been implemented in the simulated virtual environment such as water simulation, raining or gas simulation.

In the simulation, however, few tasks need to be concurrently simulated with the virtual environment. Among these tasks are lighting, shadowing, texturing, culling, and collision detection. These tasks are considered as a realistic add-on whereas in order to maintain the real-time virtual environment, these tasks need to be implemented in order to make the virtual environment becoming more realistic. If it is not, the simulated environment might become dull and cannot be realistic enough to attract the other people depending on what the application is targeted.

The realistic effect that has been put into virtual environment in order to make the simulation more interesting is collision detection. Many other fields such as networking and medical where it

involves component intersection checking have used the term collision detection itself. Collision detection is the critical component for simulated environment as it has been used to measure the realism between intersecting object in motion. Most researchers refer collision detection as an important tool for robotic, medical simulation, and computer games. Real-time simulation always tries to simulate the collision detection process as realistic as possible and thus the researchers have come out with numerous techniques in order to discuss between two or more intersected object.

The collision detection consists of two parts, which are discrete collision detection and continuous collision detection. Compared to continuous collision detection (CCD), discrete collision detection (DCD) is much faster in term of collision checking while CCD is more accurate. These two attributes cannot share the same advantages as increasing speed will eventually decreasing the accuracy of collision detection.

2 RELATED WORKS

Significant amount of studies have revealed that collision detection between two objects can be divided into two phases, which are broad phase and followed by narrow phase. [1] suggested that broad phase stands for the first phase of detecting object interference by checking which objects has collided. Next, narrow phase will be carried out to determine the exact collisions of both objects and which parts of this pairs has collided with detailed information.

Apparently, there are many types of algorithm to detect object interference in virtual environment that can be used in urban simulation. According to [1], these algorithms are; feature-based algorithms, simplex based algorithms, image-space based algorithms, volume based algorithms and spatial data structures such as BVH[2-4] and space subdivision.

Feature-based algorithms intend to work directly with the primitives of the objects. Image space based algorithm is computed by image-based occlusion queries that usually implement on the graphics hardware (GPU). Volume based algorithms seem to work just like an image space based algorithm. However, for simplex based algorithm, it uses only the vertex of corresponding object information in order to construct a sequence of convex hulls [1]. One of the most popular simplex-based algorithms is GJK (Gilbert-Johnson-Keerthi) that becomes one of the most effective methods for determining intersection between two polyhedral [5] . In 1994, [6] presented exact collision detection to be used in large-scaled environments. Algorithm presented by [7, 8] used two types of Axis-Aligned Bounding-Boxes (AABB) which is fixed size boxes and dynamically-resized bounding boxes (dynamic boxes). They used Voronoi diagram to find a closest feature pairs. Here, they characterized the environments by the objects in motion and the complexity of the models. Regular virtual environment may require the simulation to give user satisfaction of being able to navigate through the virtual environment but that does not apply to the large-scaled environment such as urban simulation that has thousands of objects in virtual world. Hence, performing accurate collision detection may consume long time just to check possible intersection area within objects in urban simulation. Thus leaving the

only choice is to make sure that the collision detection technique work effectively.

There are another two types of collision detection method that widely been used which are BVH and space subdivision. Space subdivision intends to divide the spaces into small parts called cells but it not widely used for accurate collision detection. Example of space subdivision research can be found here [9, 10]. Bounding volume hierarchies provides more efficient technique using bounding-volumes that provides smaller and tighter hierarchies comparing to space subdivision [11-15].

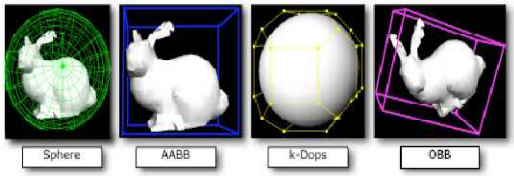

Figure 1 Examples of common BVs as described in [16].

3 BOUNDING-VOLUME HIERARCHIES

Bounding-Volume Hierarchies (BVH) is the most common use hierarchical representation in virtual environment world. Compared to the space subdivision technique that divide the space, BVH is use to split the 3D object into several parts depending on their types of tree. BVH has also widely used in culling system, collision detection system and geometry refined system. It also can be built using various type of Bounding-Volume (BV) where the complexity of BV depends on what the application is targeted. Figure 1 shows a Bounding-Volume type.

In general, BVH consists of root, nodes and branches that could be represented as a tree structure. The number of nodes is usually depending on the type of tree such as binary tree, quad tree, and oct-tree. For binary tree, the root of the tree is divided into two parts which is right node and left node. The last part of the tree is called leaf node.

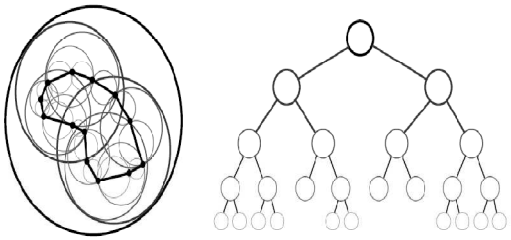

Figure 2 The left hand side image shows a BVH with Sphere BV while on the right hand side image, shows unbalanced hierarchical form using binary type hierarchy.

The BVH works by first enclosed the object with one large BV. Then, by using splitting algorithm to properly divide the object according to the type of the tree (for example, binary tree), the root has two more nodes that each node has another BV that enclosed each node. Then the splitting continues until stopping criteria is met. The stopping criteria must be set up in order to make the tree construction stopped the splitting process and avoiding infinite looping. Common stopping criteria set up by researcher are stopping when the node only has very minimum triangle or single triangle, under targeted time, or when the tree cannot be evenly divided into two parts. All these factors become the main reason why the BVH construction is important for the simulation. Bad construction could potentially get the simulation in the infinite loop of binary search. The construction process is usually done in pre-processing time except for deformable bodies simulation where BVH is need to be rebuilt after the object

has changes in their shape due to force or impact.Example of bounding-volume hierarchy is shown in Figure 2.

The unique ability for BVH to detect potential collision is by searching non-colliding pairs from the hierarchy tree. Given an example of two objects, each has their own BVH tree, the collision will not be detected if their root is not intersected and thus we eliminate the search for the tree. This process called broad phase collision detection where the first time of contact need to be detects first. However, the narrow phase collision detection will be carried out once the root of each BVH tree has come into contact and the search for child nodes will be running. It recursively checks with their child nodes until it found the precise contact between objects.

3.1 Bounding-Volume

Bounding-Volume (BV) is an important part of BVH construction. Numerous BV have been developed in the past in order to minimize the computational cost of performing collision detection. Instead of using primitive-primitive checking between intersected 3D objects, BV helps to speed up the process by enclosing bunch of triangles into single BV before proceed with collision checking. This is to reduce the possibility of eliminating set of triangles that does not intersect.

At the present time, there are several famous BVs such as spheres [17], Axis Aligned Bounding Box (AABB) [18-20], Oriented Bounding Box (OBB) [11, 20, 21], Discrete Oriented Polytope (k-DOP) [22], Oriented Convex Polyhedra [16], and hybrid combination BV [1]. Most large scale 3D simulations used bounding box because of the simplicity, require less storage, fast response of collision, and easy to implement [23]. Figure 2 illustrates most commonly used bounding volume.

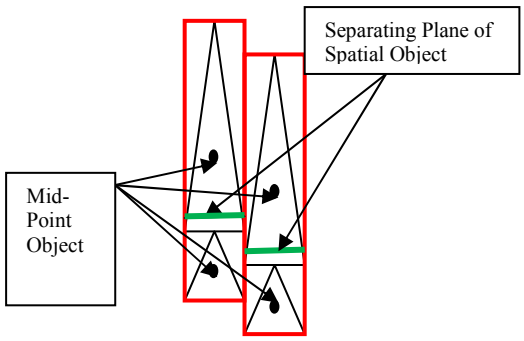

Figure 3 An example on how the SOMS rule split the object triangle into two parts using the spatial midpoint between midpoints of two triangles which will become their separating plane.

4 SPATIAL OBJECT MEDIAN ALGORITHM

Initially, the construction of BVH starts by constructing the root of the BVH which is a big BV. In this experiment, we have chosen Axis-Aligned Bounding-Box (AABB) as it produces fast and nearly accurate collision detection. Various of researchers has already chosen AABB for their researches process due to its design for the collision process.

Once we have constructed the BV, the process of BVH creation starts by finding the midpoint of the corresponding AABB by using the longest axes to split. Cartesian coordinates consists of variable x,y, and z for 3D object. The first step for midpoint calculation is using all the vertex point for corresponding AABB to find the longest extend for the distance between two points. By finding the longest distance between two points, we

have determined the respective axis for us to split.

The second step for midpoint calculation is proceed by finding the midpoint for each triangle for the object. This is true for all BVH construction as hierarchical representation required the object BV to be divided into several parts using midpoin of the triangle. Each of the midpoint will be stored into temporary variable that is declared as an array or database.

The differences between our algoritm with the common spatial median splitting algorithm is that we have used the midpoint itself to calculate new splitting axes.From Figure 3, it shows that new separating axis plane located between those two triangles midpoints. Thus, it could properly assign left and right nodes and store the corresponding triangle until it could have one BV one triangle. However for SOMS, it creates temporary spatial median of object median thus creating new median point.

Balance BVH tree is more efficient and fast when performing collision detection compared to unbalance BVH tree. Instead of faster construction using SOMS technique compared to Spatial Median, balance BVH tree helps reducing the potential of performing primitive-primitive testing on the earlier phase of detection. For example, given a node A with 40 triangles (using Spatial Median splitting rules and stop at level 7) and a node B with 20 triangles (using SOMS rules and stop at level 8). For each triangle of node A, it checks 40 times with the other object triangle. If the other object has 40 triangles too, it means that 40 x 40 tests must be done. However if we increase the level of BVH tree make it more balance just like node B with 20 triangles. It only needs to check 20 x 20 times for collision given each object while the construction time is similar. Although it is one level high compared to Spatial Median splitting rules technique, SOMS needs to perform only one test to move into the next BVH tree level for collision checking.

5 TREE CONSTRUCTION TIME TEST FOR FIXED BALANCED LEVEL OF SPATIAL MEDIAN

Figure 4 shows that SOMS technique is faster than Spatial Median technique in term of construction time. SOMS is able to construct balanced BVH at fixed level 7 while Spatial Median become unbalanced when it reaches level 7. By analyzing the Figure 4, we should notice that the BVH tree has been constructed multiple times (1000 times) and the average of these values is calculated. Hence, if the object consists hundreds of thousands polygon, we can concluded that the construction time for real application is dropped when using SOMS technique compare to Spatial Median technique. Apart from that, the Figure 4 explained the number of leaf nodes for both techniques are not in the same BV size.

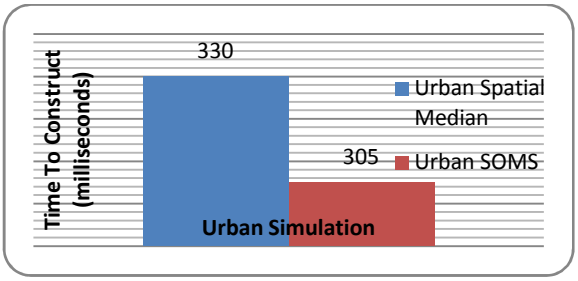

Figure 3 Average times to construct 1000 times BVH for 3DS Urban Simulation using Spatial Median and SOMS method. The Level of BVH is

fixed at level 6 (Spatial Median Balanced BVH).

6 TREE CONSTRUCTION TIME TEST FOR FIXED BALANCED LEVEL OF SOMS

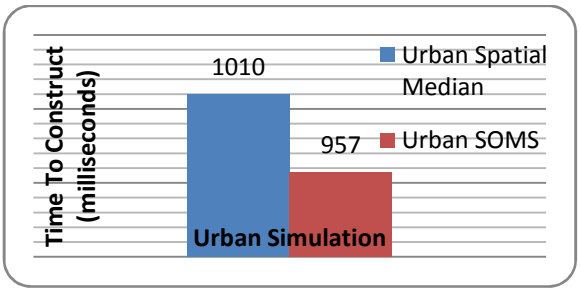

Figure 4 Average times to construct 1000 times BVH for 3DS Urban Simulation using Spatial Median and SOMS method. The Level of BVH is fixed at level 7 (SOMS Balanced BVH).

In this experiment, Spatial Median technique generated less nodes compared to SOMS technique. SOMS BVH tree needs to produce balanced BVH level tree thus level 7 BVH tree is supposed to generate 2^7 = 128 nodes. Each node is supposed to fill with triangles. However for Spatial Median technique, BVH that is generated has fewer nodes than the actual total nodes (less than 128 nodes). Hence, this proof that even though the results showed that SOMS much faster than Spatial Median, SOMS still be able to construct faster BVH tree with 128 nodes compared to Spatial Median technique.

In previous statement, we can see how number of triangles for certain objects is really important for SOMS technique to press its advantage. From the previous figure below calculation of Total Time in Minute for BVH tree construction, SOMS technique proved that for 1000 times construction test, it consumed 15 minutes and 57 seconds. Meanwhile for Spatial Median technique, the testing was running for 16 minutes and 50 seconds. The formula that we introduced here is to calculate the total time to construct 1000 times of BVH tree. It is our measurement to find BVH tree construction total time. From the testing that we run, we concluded that for any object that has hundreds of thousands polygon, SOMS be able to perform better than Spatial Median technique.

$$\text{Total Time in Minute (SOMS)} = \frac{957 \times 1000\ times}{1000\ milliseconds \times 60} = 15\ minutes\ and\ 57\ seconds$$

$$\text{Total Time in Minute (Spatial Median Splitting)} = \frac{1010 \times 1000\ times}{1000\ milliseconds\ \times\ 60} = 16\ minutes\ and\ 50\ seconds$$

From the calculation, it showed that SOMS technique produce a better BVH tree by achieving relatively less time compared to Spatial Median technique for just 5000 triangles. As most of the complex environments consisting hundreds of thousands polygon, the number of time taken to construct BVH tree is going to increase. Hence, the outcome of this testing is to measure how efficient BVH tree construction if the complex environments consist a lot of triangles.

7CONCLUSIONS AND FUTURE WORK

This technique intends to help reducing time to construct BVH while creating more balanced level or the tree reducing the use of heuristic determination. Since previous researchers used some heuristic

to continue to split their tree, SOMS intends to reduce time to use heuristic for the node determination. This could benefit the real time construction when the real application of SOMS can be applied in deformable models instead of rigid bodies in future work

REFERENCES

[1] 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.

[2] H. A. Sulaiman and A. Bade, "Continuous Collision Detection for Virtual Environments: A Walkthrough of Techniques " *electronic Journal of Computer Science and Information Technology,* vol. 3, 2011.

[3] 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.

[4] N. M. Suaib, A. Bade, D. Mohamad, and H. A. Sulaiman, "On Faster Bounding Volume Hierarchy Construction for Avatar Collision Detection," in *International Conference on Computer Technology and Development, 2009. ICCTD '09*, 2009, pp. 430-434.

[5] 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.

[6] J. D. Cohen, M. C. Lin, D. Manocha, and M. Ponamgi, "I-COLLIDE: an interactive and exact collision detection system for large-scale environments," presented at the Proceedings of the 1995 symposium on Interactive 3D graphics, Monterey, California, United States, 1995.

[7] Y. XianYi and C. Guo, "Human-Computer Interaction Design in Product Design," in *Education Technology and Computer Science, 2009. ETCS '09. First International Workshop on*, 2009, pp. 437-439.

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

[9] A. Perez, C. E. D'Attellis, M. Rapacioli, G. A. Hirchoren, and V. Flores, "Analyzing blood cell concentration as a stochastic process," *Engineering in Medicine and Biology Magazine, IEEE,* vol. 20, pp. 170-175, 2001.

[10] J. Bittner, P. Wonka, and M. Wimmer, "Visibility preprocessing for urban scenes using line space subdivision," in *Pacific Graphics 2001 (Ninth Pacific Conference on Computer Graphics and Applications*, 2001, pp. 276-284.

[11] 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.

[12] L. Aiping and Z. Qinglin, "Correlations Among Cartilage Erosion, IgA Level, Red Blood

Cell and Platelet Counts in 436 Rheumatoid Arthritis Patients with Path Analysis," in *Bioinformatics and Biomedical Engineering , 2009. ICBBE 2009. 3rd International Conference on*, 2009, pp. 1-3.

[13] D. O. Tuft, "A System For Collision Detection Between Deformable Models Built On Axis Aligned Bounding Boxes And Gpu Based Culling," Master of Science, Department of Computer Science, Brigham Young University, Brigham, 2007.

[14] A. Nguyen, "IMPLICIT BOUNDING VOLUMES AND BOUNDING VOLUME HIERARCHIES," Doctor of Philosophy, Stanford University, 2006.

[15] A. Sanna and M. Milani, "CDFast: an Algorithm Combining Different Bounding Volume Strategies for Real Time Collision Detection," in *SCI*, 2004, pp. 144-149.

[16] 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.

[17] L. Liu, Z.-q. Wang, and S.-h. Xia, "A Volumetric Bounding Volume Hierarchy for Collision Detection," in *10th IEEE International Conference on Computer-Aided Design and Computer Graphics, 2007* 2007, pp. 485-488.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

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