A Survey of Continuous Collision Detection

Quan Nie*[1], Yingfeng Zhao[2], Li Xu[3], Bin Li[4]
National Key Laboratory of Science and Technology on Vacuum Electronics
Academy of Computer Simulation Technology, University of Electronic Science and Technology of China
NO.4, Section 2, North Jian she Road, Chengdu, 610054, China
*[1]18795863794@163.com; [2]zhaoyf@uestc.edu.cn; [3]lixu@uestc.edu.cn; [4]libin@uestc.edu.cn.

***Abstract*—Continuous collision detection (CCD) is a key technology in the field of virtual surgery, cloth simulation and robot motion planning. It can accurately detect the first time of contact between objects and returns collision information such as penetration depth, friction and repulsive force, etc., have a wide range of application and important research value. By analyzing the processing framework of continuous collision detection algorithm in detail, the current research status of continuous collision detection is systematically reviewed from perspectives of two phases respectively. In broad-phase, the recent achievements of space decomposition and sweep and prune are introduced. In narrow-phase, the research status of intelligent optimization based algorithm and image-space based algorithm is illustrated. Besides, the development of bounding volume hierarchy (BVH) is analyzed and discussed. After that, the performance and innovative achievements of self-collision detection in deformable objects are summarized and analyzed. Finally, the challenges and future trends of algorithm research are pointed out.**

Keywords-Bounding volume hierarchies; Parallel computation; Analysis of Algorithms; Self-Collision Detection

I. Introduction

Continuous collision detection (CCD) is a core technology in physics-based virtual simulation, robot motion planning, virtual surgery, CAD/CAM. It is the guarantee of the authenticity, real-time performance and actual operation safety of virtual simulation process. In virtual surgery and robot motion planning, the objects to be dealt with are deformable objects such as skin and soft obstacles. In these scenes, CCD is not only to detect the interference between objects, but also to inspect the self-collision in deformable objects. The following discussion mainly takes the deformable objects in the virtual scene as the research object, and carries on the systematic review to the CCD algorithm. Over the past decade, a variety of CCD algorithms have been proposed including: algebraic equation solving [1], swept bounding volumes [2], kinetic data structures [3] and configuration space approach [4]. Among them, the first one is combined with intelligent optimization algorithm to improve the accuracy of collision detection (CD). In recent years, many scholars combine the latter three methods with GPU and propose many efficient, robust and accurate algorithms. Meanwhile, with the rapid development of computer hardware, the application of parallelization technology based on CPU and/or GPU to CCD algorithm to break the algorithm speed bottleneck has become a research hotspot and trend in the past decade. In the following review, it can be seen that many CCD algorithms improved by the rasterization capabilities of GPU to accelerate detection.

Based on the understanding of the pipeline of collision detection algorithm, this paper mainly systematically illustrates and analyzes the broad-phase and narrow-phase CCD algorithms, which is of great significance to the algorithm design and improvement. In section II, we elaborate the broad-phase algorithms. Space decomposition and sweep and prune are generally used to quickly eliminate object pairs that cannot collide, and identifies potential groups of possibly intersecting objects [5]. In section III, the narrow-phase algorithm inspects further these potential pairs and finally intersection inspect algorithms determine the exact colliding pairs, which include intelligent optimization and image-space based algorithm. Besides, several bounding volumes hierarchy (BVH) are compared and analyzed horizontally, and the development process of BVH's construction is illustrated. In section IV, the research achievements of self-collision detection in deformable objects are compared and analyzed. Section V emphasizes on the challenges and problems in collision detection technology. Finally, some research trends of collision detection technology in the future are summarized and prospected.

II. Broad-phase collision detection

A. Spatial decomposition

The spatial decomposition method divides the three-dimensional space into multiple subspaces and creates a list of objects which includes all models in subspaces. Basic spatial decomposition uses a uniform grid of cells, but a hierarchy of spatial subdivisions can also be used. Typical hierarchical structures include BSP-Trees, octrees and K-D trees [6]. Bao et al. [7] proposed an octree structure based on adaptive meshing divide and optimized coding, which was applied to the surgical training system with uneven and dense distribution of objects. This algorithm uses three times of the average side length of the AABB which wraps the model to set the octrees' element size and adopts the adaptive triangle subdivision method in the stage of accurate detection of graph elements, which eliminates many redundant tests and improves the detection efficiency. Francisco et al. [8] proposed a collision detection algorithm based on the EBP-Octree (Extended Bounding-Planes Octree), which solves the problem of CD between high-resolution polygon models. The outstanding feature of the structure is that it owns near real-time performance when tens of millions of objects move and penetrate each other, and can adjust memory consumption according to hardware performance, which is suitable for fast and accurate intersection tests between triangles in almost any environment.

This work is supported by National Natural Science Foundation of China (Grant No. 61771105) and Fundamental Research Funds for Central Universities (Grant No：2672018ZYGX2018J037).

978-1-6654-0378-8/20/$31.00 ©2020 IEEE
DOI 10.1109/ITCA52113.2020.00061

Spatial decomposition can significantly reduce the number of feature pairs to be tested. However, the algorithms have obstacles of large memory consumption and poor stability when dealing with scenes with uneven model distribution. In addition, for the collision detection in deformable bodies, the real-time performance of updating or rebuilding the hierarchy is also a big problem.

B. Sweep and prune

Sweep and prune (SAP) has been proved an efficient strategy in the scenes of large moving objects and can ensure that broad-phase collision culling can be completed in linear time. In recent years, SAP algorithm combining kinetic data structure with graphics hardware and parallelization technology have been proposed. Wang et al. [9] applied SAP algorithm to detect the self-collision within cables. The algorithm first constructs a piecewise mathematical expression of cables. Then it traverses all discrete points of cables and implements SAP to calculate the penetration depth of cables. Finally, the interference is determined based on the ratio of the penetration depth to the sum of the cable radius. This algorithm has high accuracy and is suitable for collision detection in the process of cable routing. Capannini et al. [10] proposed an adaptive SAP algorithm based on CPU parallelism and context awareness, which is suitable for broad-phase collision detection in large-scale complex scenes. The solution reduces range query complexity by using biaxial sweep and an improved data structure. During the parallel sorting phase, temporal-coherence is exploited to quickly distribute workload to worker threads. In the pairing process, the adaptive partitioning method with good scalability is adopted to improve the algorithm throughput. Finally, experimental results show the proposed strategy lead to s a fast parallel collision-culling algorithm. However, the drawback is that exchange data frequently between the graphics device and the system CPU can lead to a lot of expenditures. Qi et al. [11] proposed a CCD algorithm based on kinetic SAP framework and event-driven mechanism. The strategy automatically generates events to predict the object collisions to occur and presents event blocking control mechanism which can well check the potential event blocking in the collision process and improve the performance and stability of the algorithm. This algorithm has obvious advantages in time consumption and is more suitable for multi-object collision detection than the frame-based SAP methods.

III. Narrow-phase collision detection

A. Intelligent optimization based algorithms

Over the past decades, many optimization algorithms such as particle swarm optimization, neural network, linear programming and hybrid optimization algorithm have been proposed. The Intelligent optimization based collision detection is an excellent algorithm for narrow-phase collision detection. It has higher accuracy than other algorithms, but the problems are high time consumption and premature iterative convergence. Jin et al. [12] proposed a new algorithm aiming at the real-time and accuracy of collision detection between deformable objects. Based on random collision detection, this algorithm combine particle swarm optimization (PSO) with differential evolution (DDE) algorithm to improves the global searching ability and speeds up the convergence speed of the algorithm. Compared with other algorithms, the hybrid algorithm has higher accuracy. However, the intelligent algorithm increase the computation amount and time consumption. Qu et al. [13] introduces the optimization operator to re-initialize the iterative near-stagnation subgroup to avoid premature convergence falling into the local optimal solution. Secondly, the multi-particle swarm is fine-grained decomposed into subtasks and executed using SIMD parallel algorithm. This algorithm has obvious advantages in terms of detection efficiency and accuracy, which can meet the requirements of real-time and accuracy of collision detection in human-computer interaction. Wen et al. [14] introduced PSO algorithm to optimize the random group in two-dimensional discrete space, and then found the feature pairs of collisions. The algorithm optimizes the speed of collision detection, reduces memory consumption and improves the real-time performance of collision detection.

B. Image-space based algorithms

Francois et al. [15] proposed a collision detection and response algorithm based on image volume minimization, which used layered depth image (LDI) to represent multi-layer geometric structure. GPU is used to render the polygon of intersecting surface into LDIs in three orthogonal directions and calculate the gradient of intersecting volume at the vertex. Then the contact force is calculated according to pixel depth. This method does not require geometric preprocessing, so it is effective for both elastomers and rigid bodies. However, the accuracy of this method influenced by the resolution of LDI and buffer. Besides, collide may be missed in some cases. Du et al. [16] proposed a parallel CCD algorithm for N-body scenario to solve the problem of performance gain be limited on a single processor. The algorithm firstly distributes the workloads evenly on the nodes of high-performance GPU cluster by spatial decomposition method, and then uses various GPU-based efficient eliminate methods such as sphere bounding volume and parallel SAP to get the colliding pairs. The strategy has better time efficiency than the existing serial/share-memory CCD methods and is more suitable for large-scale distributed simulation scenarios.

The problem of load balancing between CPU and GPU is a bottleneck restricting the development of image-space based algorithm. Tayyub [17] proposed a successive approximation method to evaluate the optimal load division ratio. The heterogeneous processing based on GPU-CPU platform is adopted to improve the efficiency of collision detection. The effectiveness of this method is proved in the actual benchmark.

C. Bounding volume hierarchy

The most widely used bounding volumes in CCD include axis-aligned bounding volumes (AABB) [18], spherical bounding volumes [19], oriented bounding volumes (OBB) [20], and discrete orientation polytopes (K-DOP) [21], etc. With the extensive development of the research, various forms of bounding volumes have been proposed successively, such as ellipsoid, cylinder, spherical shell [22], etc. Table. 1 shows the performance comparison results of several bounding volumes under general conditions [23], and Fig. 1 shows the form of these bounding boxes on a two-dimensional plane.

TABLE I. COMPARISON OF SEVERAL BVS (1 IS OPTIMAL)

Bounding volume	Simplicity	Memory usage	Complexity of intersection tests	Tightness	Applicability of deformable body
AABB	1	2	2	5	1
Spheres	2	1	1	6	3
Cylinder	3	4	3	4	4
Ellipsoid	4	3	4	3	5
OBB	6	5	6	2	6
K-DOP	5	6	5	1	2

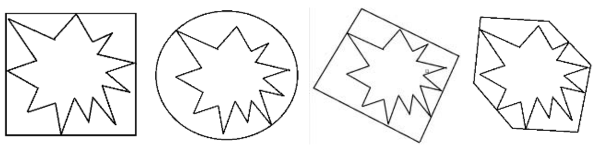

Figure 1. Two-dimensional forms of BVs

In the past, due to hardware architecture limitations, most algorithms relied on serial algorithms running on the CPU to construct and update the BVH tree, and then did the actual collision queries on the GPU. In recent years, researchers have begun to combine GPU rendering with the construction of BVH. Directly constructing BVH structure on GPU can avoid the relatively expensive delay caused by frequent duplication of data structure between CPU and GPU memory space, so improve the efficiency of collision detection and reduce the algorithm complexity. Among BVH-based technologies, the most time-consuming operation is BVHs' maintenance. In large-scale dynamic scenarios, the collision detection of deformable models usually gives priority to build speed rather than quality or tightness. Lauterbach et al. [24] introduced a linear BVH (LBVH) structure which uses spatial Morton codes to simplify the BVH construction problem into node ordering problem to facilitate the division of complex objects and significantly improve the BVH's construction speed. Since its inception, LBVH has been extended many times and has inspired many build algorithms. Reference [25] extends this algorithm and proposes a surface heuristic linear BVH (HLBVH) algorithm, which reduces detection computing consumption and memory usage by utilizing the coarse-grained space-coherence of hierarchical grids. Karras [26] proposed a depth-first binary radix tree technique and constructed the whole tree within the linear time complexity. Apetrei [27] proposed a bottom-up fast construction strategy requiring only one GPU kernel to construct the hierarchy and calculate the BV's boundaries. This strategy is relatively efficient but complex, and needs to analyze the partition of the internal nodes of the BVH tree to establish the connection between the node index and the Morton code range.

Based on the spatio-temporal coherence of moving objects between successive frames, the bounding volume traverse tree (BVTT) front is proposed, which is defined as the traversal path of non-overlapping nodes on BVTT. Reference [28] first proposed an incremental method that takes advantage of spatial coherence to accelerate CD, which also known as generalized forward tracking. The algorithm was applied to a GPU-based streaming algorithm in [29] later. After that, in order to solve the problem that the model deformation will reduce the rejection efficiency and query speed, and the GPU cache efficiency is low due to the arbitrary layout of BVTT. Wang [30] proposed a fast CD scheme based on GPU, which proposed histogram sorting and parallel calculation of BVH and BVTT fronts, and BVH nodes' adaptive adjustment strategy based on update log. In the recent research, Floyd M et al. [31] proposed a method of rapidly constructing BVH trees based on GPU. The algorithm firstly realizes implicit index mapping to GPU memory based on bit transformation, and constructs binary coded BVH tree. Then, the BVH tree is reconstructed or updated rapidly in each frame through GPU scheduling, with time complexity of .This method stores only a minimum number of BVH nodes to greatly reduce memory cost. Meanwhile, this method builds BVH six times faster than the recently proposed optimal algorithm.

In addition, many Chinese scholars have improved the traditional algorithm by introducing the geometric characteristics of the model, and some achievements have been made. Sun et al. [32] calculated the local coordinate system of OBB based on triangular area weighted covariance, and applied it to the construction process of BVH to solve the bias problem of OBB. Moreover, to improve the efficiency of CD, a new coordinate system is introduced in the basic element detection stage, and the spatial triangle is projected to the two-dimensional plane to judge the intersection. Li [33] introduced centroid in material mechanics to determine the direction and position of OBB, and improved performance by adjusting the weight of BVH's nodes. Tang et al. [34] proposed an algorithm based on hybrid OBB which perform coarse-grained inspection after particle conversion and fine-grained detection after particle reduction according to the model topology, effectively reducing the detection and calculation time.

IV. SELF-COLLISION DETECTION

Deformable object refers to the clothing fabric, surgical simulation of the skin, organs and other soft or elastic material model. In the virtual simulation environment, the deformable model is an abstraction of the soft object in the real environment, which can simulate the deformation of the soft object under the action of external forces and stresses. The self-collision detection in deformable models is much more complicated than that in the rigid models, because the structural changes of the deformable models in motion, such as folding, fragmentation and deformation, require frequent reconstruction or update of the data structure used for detection. In addition, the increasing computation cost due to the deformation of objects makes the algorithm time consuming more and more. To decrease the number of pairs of intersection tests and the computational complexity, many scholars proposed to eliminate redundant pairwise BV tests to improve performance. As shown in Table 2 below, we compared and summarized the research developments from self-collision detection in recent years. These methods mainly include the fast CCD algorithm based on normal cone culling and GPU.

TABLE II. SELF-COLLISION IN DEFORMABLE MODELS

Author	Performance of algorithms	The innovation and limitation
Tang, Sean Curtis et al. [35]	● Fine scalability. ● Low storage and runtime costs. ● GPU-based serial algorithm which takes rasterization hardware to accelerate algorithms.	● Two-level Dynamic BVH which based on k-DOPs. ● High-level culling based on continuous normal cones algorithm. ● Procedural representative triangles (PRTs) and Orphan set to reduce the number of exact interference tests.
Tang, Manocha et al. [36]	● Fine scalability. ● Reduce memory overhead through improved algorithms. ● The task nodes of collision query are organized as blocks and dynamically distributed to the child threads.	● The front-based incremental algorithm applied for rigid body model is extended to self-collision detection in deformable models. ● Cache-friendly memory layouts of BVTT and the BVH.
Tang et al.[37]	● Weak scalability ● Compressed diagonal format (CDF) of matrix representation reduces the total memory footprint by 50%. ● Simulating highly detailed folds and wrinkles by GPU stream data and kernels.	● A GPU-based streaming pipeline that maps all the components include time integration, collision checking and force computation to GPU. ● Collision handling methods can't resolve all the penetrations. ● Optimized sparse matrix representation is not suitable for cloth models represented by triangular meshes.
Wang et al.[38]	● General scalability ● Normal cone combined with BVTT Front improve the size and memory overhead. ● Serial execution of a single CPU.	● Present a criterion for self-collision checking using surface normal cones, binormal cones and boundary contours of geometry. ● The algorithm is effective only when the curvature of the geometric mesh does not show high variation. ● More computational complexity and time consumption for BVHs' update.

V. EXISTING PROBLEMS AND CHALLENGES

The Existing methods have at least the following challenges.

A. Real-time collision detection of large models

At present, the mainstream collision detection algorithm is usually driven by the frame-based sample. During simulation, collision detection and frame rendering must be performed simultaneously. Although the frame-based method is easy to implement, it is difficult to meet the requirement of high refresh rate of the system display, such as the refresh frequency of the tactile interaction in a specific area up to 1 kHz [39].Therefore, how to make the algorithm meet the interaction requirements of large-scale model scenes and carry out efficient real-time collision detection has become a difficult problem.

B. Accuracy problems of CCD algorithm

Most CCD algorithms are implemented with finite-precision arithmetic which can lead to two kinds of precision problems: a false negative (FN) that occurs when the algorithm misses collision. When the algorithm classifies a non-colliding instant as a collision, false positive (FP) occur. The accuracy and real-time performance of the algorithm are always in conflict with each other. Too much accurate arithmetic operation will increase the computational load and thus reduce the speed and efficiency of CCD algorithm. Furthermore, it can be difficult to implement precise arithmetic operations or libraries on a GPU or embedded processor. But the problem is that too many false positives and false negatives will seriously influence the performance and robustness of collision detection and response calculation.

C. *Memory consumption*

BVH is the most commonly in CCD algorithm. But as a typical tree structure, it needs to store a large number of bytes in the bounding volumes' node of each layer. Especially when there are a lot of complex models in the scenario, the memory consumption is huge. On the other hand, in order to solve the state update problem of deformable objects, researchers propose a method to reconstruct BVH trees in each frame. However, most BVH-based GPU algorithms require the calculation and storage of connectivity between BVH nodes, which introduces an indirection and increases the memory overhead and construction time. Therefore, reducing redundant data storage and optimizing memory is also an urgent problem.

VI. CONCLUSION AND FUTURE WORK

Continuous collision detection is a key technology in virtual assembly environment, virtual surgery and computer animation, which directly affects the reliability and availability of simulation results. After systematically analyzing the research status and problems, the CCD is summarized and prospected as follows.

Researchers put forward efficient hierarchical structure such as BVTT (and BVTT front) that based on the spatio-temporal coherence between successive frames and the node sorting algorithms. However, the problem about how to quickly build high-quality BVH and effectively cut memory overhead is still a need to further discuss.

Many scholars have begun to improve various basic CD algorithms to absorb each other's advantages so as to research many efficient, robust and accurate algorithms. With the continuous development of computer hardware technology, it will become a general trend to use parallelization technology to accelerate CCD algorithm.

There are thousands of objects in the large-scale virtual scene, and the three-dimensional model of these objects is composed of millions of complex features. The real-time self-collision detection between deformable objects in that scene is a big problem. GPU-based algorithms have inherent limitations which is GPU memory capacity is limited and not all computers have graphics hardware, so it is difficult to meet the high refresh frequency of tactile interaction in this scene. Improving and extending the incremental algorithm based on event-driven may be a breakthrough point. In addition, it is also worth further research and discussion to break through the performance limitation by balancing the load distribution in the high-performance GPU and CPU cluster.

REFERENCES

[1] X. Jia, Y. K.Choi, B. Mourrain, W.Wang. An algebraic approach to continuous collision detection for ellipsoids[J]. Computer Aided Geometric Design, pp. 164–176, 2011.

[2] J. A. Corrales, F. A. Candelas, F. Torres. Safe human-robot interaction based on dynamic sphere-swept line bounding volumes[J]. Robotics and Computer Integrated Manufacturing, pp. 177-185, 2011.

[3] D. Kirkpatrick, J. Snoeyink, B. Speckmann. Kinetic collision detection for simple polygons[J]. International Journal of Computational Geometry & Applications, pp. 3-27, 2011.

[4] Y. Lee, E. Behar, J. M. Lien, Y J. Kim. Continuous penetration depth computation for rigid models using dynamic minkowski sums[J]. Computer Aided Design, pp. 14-25, 2016.

[5] W. René. New geometric data structures for collision detection and haptics[J]. Journal of Applied Ichthyology, pp. 348-353, 2013.

[6] R. R. Man, D. S. Zhou, Q. Zhang. A survey of collision detection[J]. Applied Mechanics & Materials, pp. 360-363, 2014.

[7] Y. D. Bao, D. M. W. Adaptive subdivision and Optimization of coded octree collision detection algorithm[J]. Journal of Shanghai Jiaotong University, pp. 1114-1122, 2015.

[8] F. J. Melero, A. Aguilera, F. R.Francisco. Fast collision detection between high resolution polygonal models[J]. Computers & Graphics, pp.83: 97-106, 2019.

[9] F. L. Wang, Y. Guo, W H. Liao, S H. Huang. Cable collision detection technology based on distance field and Sweep cutting algorithm[J]. Computer Engineering and Application, pp. 27-34, 2017.

[10] G. Capannini, T. Larsson. Adaptive collision culling for massive simulations by a parallel and context-aware sweep and prune algorithm[J]. IEEE transactions on visualization and computer graphics, pp. 2064 – 2077, 2017.

[11] B. Qi, M. Pang. An enhanced sweep and prune algorithm for multi-body continuous collision detection[J]. The Visual Computer, pp.1503-1515, 2019.

[12] Y X. Jin, C. Ren et al. Research on Deformed Body Collision Detection Algorithm Based on Intelligent Algorithm[J]. Computer Engineering and Applications, pp.53(19): 130-135, 2017.

[13] H. Qu, Y. Zhao, A. Qin. Fast collision detection algorithm based on optimization operator[J]. Journal of Jilin University, pp. 275-280, 2017.

[14] C. Wen, W. Xu, W Q. Tang. Research on Collision Detection Algorithm Combining Effective Constraints OBB and PSO[J]. Modern Electronics Technique, pp.43(13):95-98, 2020.

[15] F. Faure, S. Barbier, A. Jérémie, F. Falipou. Image-based Collision Detection and Response between Arbitrary Volumetric Objects[C]. Eurographics/acm Siggraph Symposium on Computer Animation. ACM, 2008.

[16] P. Du, E. S.Liu, T .Suzumura. Parallel continuous collision detection for high-performance GPU cluster[C]. Acm Siggraph Symposium on Interactive 3d Graphics & Games. ACM, 2017.

[17] M. Tayyub, & G. N. Khan. HETEROGENEOUS CPU-GPU IMPLEMENTATION OF COLLISION DETECTION[C]. International Conference on Applied Computing, 2019.

[18] X. R. Wang , M. Wang, & C. G. Li. Research of collision detection algorithms based on aabb[J]. Computer Engineering & ence. 2010.

[19] X. Y. Qu, L. Ma, & C. Z. Yao. Research of collision detection algorithm based on hybrid bounding box for complex environment[C]. 2016 International Conference on Integrated Circuits and Microsystems (ICICM). IEEE. 2016.

[20] S. Gottschalk, M. Lin, D.Manocha. OBB-Tree: A hierarchical structure for rapid interference detection[C]. Proceeding of ACM SIGGRAPH 1996. Dallas, pp. 171-180, 1996.

[21] J. T. Klosowski, M. Held, J. S. B. Mitchell, H. Sowizral, K. Zikan. Efficient collision detection using bounding volume hierarchies of k-dops[J]. IEEE Transactions on Visualization and Computer Graphics, pp. 21-36, 1997.

[22] S. Dinas, M. José. A literature review of bounding volumes hierarchy focused on collision detection[J]. Ingeniería Y Competitividad Revista Científica Y Tecnológica, pp. 49-62, 2015.

[23] Y. S. Zhou, G F. Ding, & M. H. Xu, Y. He. A survey of real-time collision detection algorithms[J]. Application Research of Computers, pp.8-12, 2008.

[24] C. Lauterbach, M. Garland, S., & Sengupta et al. Fast bvh construction on gpus[J]. Computer Graphics Forum, pp. 28(2):375-384, 2009.

[25] J. Pantaleoni and D. Luebke. HLBVH: hierarchical LBVH construction for real-time ray tracing of dynamic geometry[C].In Proceedings of the Conference on High Performance Graphics. ACM, 2010.

[26] T. Karras. Maximizing Parallelism in the Construction of BVHs, Octrees, and k-d Trees[C]. Acm Siggraph. ACM, 2012.

[27] A. Ciprian. Fast and simple agglomerative lbvh construction[J]. Computer Graphics & Visual Computing,2014.

[28] T. Y. Li, & J. S. Chen. Incremental 3d collision detection with hierarchical data structures[C]. In Proceedings of the ACM Symposium on Virtual reality software and technology, pp 139-144, 1999.

[29] P. Du, J. Y. Zhao, W. B. Pan, & Y. G. Wang. Gpu accelerated real-time collision handling in virtual disassembly[J]. Journal of Computer Science and Technology, 30(003): 511-518, 2015.

[30] X .L. Wang , M. Tang , M. Dinesh , R. F. Tong. Efficient BVH‐based Collision Detection Scheme with Ordering and Restructuring[J]. Computer Graphics Forum, 37(2):227-237, 2018.

[31] F. M. Chitalu , C. Dubach , T .Komura. Binary Ostensibly implicit Trees for Fast Collision Detection[J]. Computer Graphics Forum. pp. 509-521, 2020.

[32] J .R. Sun, X. M. Lu. Collision detection optimization algorithm based on hybrid bounding box and triangle intersection[J]. Computer Engineering and Applications, 54(19):204-209, 2018.

[33] Y. H. Li, Z. Y. Wang. Improved collision algorithm based on hybrid hierarchical bounding box[J]. Journal of East China Jiaotong University, 036(006):112-118, 2019.

[34] H. Y. Tang, J. Hou, T. T. Hou. Hybrid collision detection algorithm based on particle transformation and bounding box[J]. Journal of Harbin Engineering University, 1695-1701, 2018.

[35] M. Tang, S. Curtis, S. E. Yoon, D. Manocha. ICCD: Interactive continuous collision detection between deformable models using connectivity-based culling[J]. IEEE Transactions on Visualization and Computer Graphics, pp. 544-557, 2009.

[36] M. Tang , D. Manocha, R. Tong. MCCD: Multi-core collision detection between deformable models using front-based decomposition[J]. Graphical Models, pp. 72:7-23, 2010.

[37] M. Tang, R. Tong , R. Narain, C, Meng, D, Manocha. A GPU-based Streaming Algorithm for High-Resolution Cloth Simulation[J]. Computer Graphics Forum, pp. 21-30, 2013.

[38] T Wang, M. Tang, Z. Wang, R. Tong. Accurate Self-Collision Detection Using Enhanced Dual-Cone Method[J]. Computers & Graphics, pp. 70-79, 2018.

[39] S. Rasool , A .Sourin. Real-time haptic interaction with RGBD video streams[M]. Springer-Verlag New York, Inc. 2016.