A Quad Tree Based Self-collision Detection Method for Cloth Simulation

Bing He
State Key Laboratory of Virtual Reality Technology and Systems of BeiHang University
Email: hebing@vrlab.buaa.edu.cn

Liu Cheng
State Key Laboratory of Virtual Reality Technology and Systems of BeiHang University
Email : chengliu@139.com

***Abstract*—Regarding the self-collision detection efficiency during cloth simulation, this paper presents a quad tree based self-collision detection method. In this paper, we construct a quad bounding box tree for cloth according to the spatial location of its geometric primitives from top to down. And it divides the self-collision detection process into two stages. During the first rough detection stage, with normal cone method and detection of distance between triangle pairs' centroids, we reduce triangles pairs for accurate elements intersection tests; during the accurate intersection stage, we filter out point-triangle pairs which impossible intersect using ipsilateral determination criteria rapidly. The final experimental result shows that the above optimization may greatly reduce the calculation workload of intersection detection and effectively improve the real-time while ensuring high verisimilitude.**

***Index Terms*—Cloth Simulation, Fabric Simulation, Self-collision Detection**

I. INTRODUCTION

During the cloth simulation, the cloth, expect for collision with its surrounding, will collide with itself as well, i.e., self-collision. In order to ensure the reality of simulation, it is necessary to detect these collisions and give responses timely; otherwise, penetration will occur and destroy the reality.

Self-collision process includes: Collision detection and collision response. Collision detection aims at finding out collision and calculating relevant parameters; while collision response aims at making the collided objects move correctly after collision according to the collision point and other parameters to reflect the real dynamic effect.

The existing space-based collision detection algorithm is basically divided into two categories: Space decomposition and hierarchical bounding box. The kernel of the two is to improve the efficiency of collision detection by reducing the number of elements to be detected.

Manuscript received July 20, 2009; revised Sep 10, 2009; accepted Sep 23, 2009.
Supported by the National High-Tech Research and Development Plan of China under Grant No. 2007AA01Z337

Cloth simulation calls for real-time. The speed of self-collision detection and response may directly influence simulation speed. Therefore, it is essential to design an outstanding detection and response algorithm with both reality and real-time. David Baraff [1] analyzed the collisions of fabric model lattice in each step and made corresponding responses. His method avoids the occurrence of some disorderly structures resulted by mistakes, which may further influence the process of next collision. Mathieu Desbrun [2] attached a layer of implicit surface onto the basic structure of objects to simulate some specific objects. It can rapidly generate accurate contact surface and local deformation. Doug L. James[3] and others mainly researched in interactive simulation and physical model with high creditability and proposed a pre-computation method for deformable data-drive model. Robert Bridson [4] and others focused on collision processing in simulation of models with friction characteristics. They proposed a fast algorithm to process fabric collision issue, including collision with rigid objects and self-collision. There method owns good scalability and robustness. Based on Bridson's research, David Harmon [5] and others proposed a robust algorithm processing complicated and simultaneous collisions. Andrew Selle [6] and others researched in fabric simulation with high resolution and proposed a history-based collision detection frame. Recently in paper [7], they present an interactive algorithm for continuous collision detection between deformable models. Like ours, many researchers focused on how to remove redundant elementary tests to improve the culling efficiency in cloth self-collision detection [7, 8, 9].

This paper divides the self-collision detection process into two stages: Rough detection and accurate intersection detection, as shown in Figure 1.

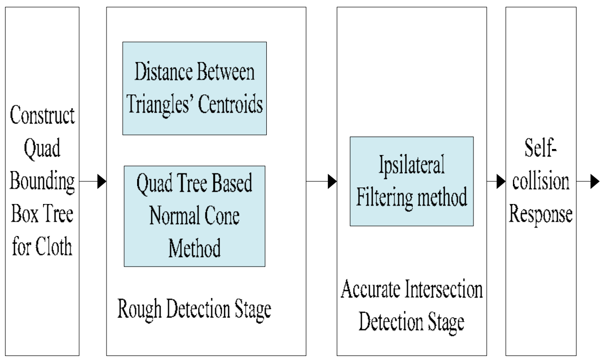

Construct Quad Bounding Box Tree for Cloth
Distance Between Triangles' Centroids
Quad Tree Based Normal Cone Method
Rough Detection Stage
Ipsilateral Filtering method
Accurate Intersection Detection Stage
Self-collision Response

Figure 1. Self-collision Detection and Response Process

Firstly, we construct a quad tree with compact structure according to the spatial location of fabric geometric primitives from top to down. Then During the rough detection stage, we use distance between triangles' centroids and quad tree based normal cone method to eliminate tri-patch pairs which impossible intersect, accordingly reduce the calculation workload for accurate elements intersection detection stage. During accurate intersection detection stage, an ipsilateral filtering method is given to eliminate unnecessary edge-face intersection tests and thus improve the efficiency. Finally, we do self-collision response to get right effect of simulation.

II. Quad Tree and Bounding Box

Fabric structure is quite complex, thus the calculation workload is huge to detect the collision of two objects. In order to improve the efficiency, some scholars adopt axial bounding box (AABB) tree to eliminate the geometric primitive pairs which impossibly intersect with each other as early as possible during the search process, thus speed up the detection.

Our Fabric model is consisted of many geometric sub-parts. Bounding boxes of them can be obtained. And these sub-parts can be divided into smaller parts recursively, thus the bounding boxes of the fabric and its sub-parts can form a tree shape hierarchical structure. The root of the tree is the bounding box bounding the whole fabric, the leaf node is the bounding box bounding geometric primitives (triangles or tri-patches), and the middle node corresponds to the bounding box of each level of sub-parts.

The traditional collision detection method adopts binary tree to establish a hierarchical bounding box tree and the leaf node includes only one primitive. Though this method is easy to understand and realize, the number of leaf node is huge, which will lead the height of tree to increase rapidly. On one hand, this will make very big consumption of memory; on the other hand, this will reduce efficiency. In addition, the traditional method does not consider the spatial location relation of geometric primitives during the construction process of bounding tree from bottom to up, which may lead to relatively loose hierarchical bounding box structure to some extent and impact the efficiency of collision detection accordingly.

Based on quad tree we construct a hierarchical bounding structure for fabric from top to down recursively, i.e., make 4*4 iterated partition for fabric according to its planar structure until tri-patches are reached. The division of fabric is shown as Figure 2.

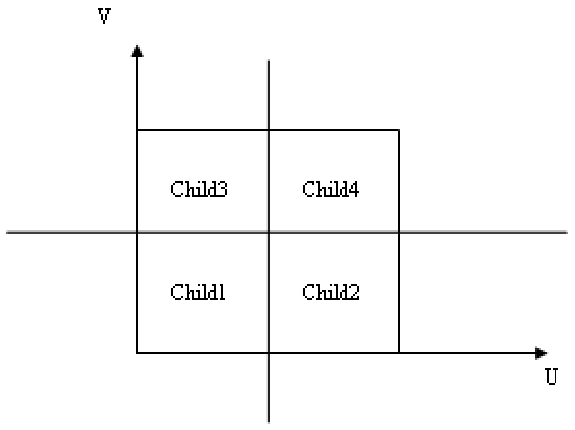


Figure 2. Fabric Quad Tree Division

The leaf nodes of this tree are tri-patches. Data information of the bounding box shall be recorded for each node. Nodes located on layer n-1 only have two leaf nodes (trangles). The formed quad bounding box tree is shown as Figure 3.

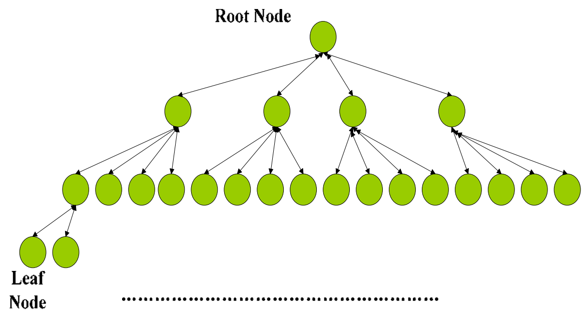


Figure 3. Quad Bounding Box Tree

Within each time step, the cone apex angle and bounding box information of each node shall be updated. We adopt post-order traversal method for updating.

In case of the same number of particles, compared to binary tree, quad tree is smaller in depth, so the recursion executed by collision detection is relatively less. Supposing the number of particles is u*u, the depth of binary tree is log $2(2*(u-1)^2)$, while the depth of quad tree is only log 2(u-1), nearly half in difference. In addition, collision detection algorithm adopts post-order traversal method, similar to the updating process.

III. Quad Tree Based Normal Vector Cone Method

The thinking of normal vector cone method originates from curvature based self-collision detection algorithm. The basic concept of curvature based heuristic method self-collision detection method is as follows:

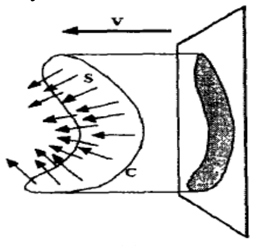
1）We can find a vector V，the dot product of V and normal vector of any primitive on surface S is larger than 0;

2）Project the external contour line C of surface S onto the plane which is vertical with V. If no self-intersection occurs, self collision will not happen on the surface, as shown in Figure 4 (a).

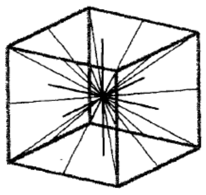
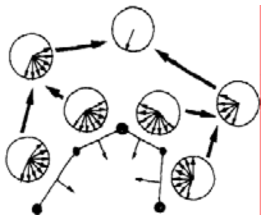
The found vector V can be processed according to the following method:

Direction sampling: Take direction sampling in unit cube, generally 26 directions shall be sampled, i.e., 26 directions connecting cube center and its apex, middle point on side, and center of surface, as shown in Figure 4(b). If among the 26 directions, there exists any direction which dot product with the normal vector of all primitives is larger than 0, no self-collision will occur on this surface. This is a process integrating continuous intersection, as shown in Figure 4 (c).

After having found V, we project the surface contour line onto the surface vertical to V and judge if the contour line is self-intersected after projection. This process is rather complex and will impact the speed in actual simulation. Thus this stage does not appear in practical application generally.



(a)

(b) (c)

(a) Projection of External Contour
(b) Direction Sampling
(c) V Solution of Direction

Figure 4. Curvature Based Heuristic Method Algorithm

Based on the design concept of curvature method, a mutated curvature method appears: Normal vector cone method. In order to detect the self-collision of fabric, Provot [10] analyzed some triangulated fabric by using the thinking of curvature method in his works. Each triangle element of fabric corresponds to a normal vector, move these normal vectors to one point and then enclose in an enough small cone (as shown in Figure 5, normal vector cone includes all normal vectors of primitives on some surface). If the apex angle of cone α<π/2, self-collision will not appeal in this area.

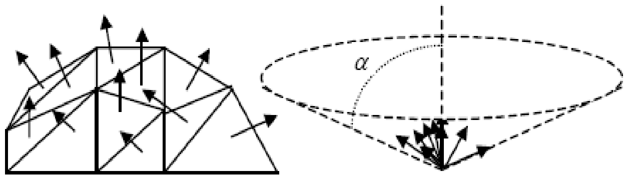


Figure 5. Cone Area Map Formed by Normal Vectors of Triangles on Fabric Surface

In the hierarchical bounding box tree of fabric, each node indicates a continuous area. We add a α attribute for each node to indicate α angle in this area. The α value of each node can be obtained through bottom-to-up traversal.

This paper adopts quad bounding box tree, so during the process of traversal solution, divide four child nodes into two groups. Firstly, solve the combined α values of two sub-nodes in each group respectively and record as α_1, α_2, and then combine α_1 and α_2, thus α value of father node of the four child nodes is obtained.

Supposing the angle between corresponding cone axes of two sub-nodes is β, the combined α value can be determined by α_1 and α_2 of two sub-nodes and β (as shown in Figure 6), as in

$$\alpha = \beta / 2 + max(\alpha_1, \alpha_2) . \quad (1)$$

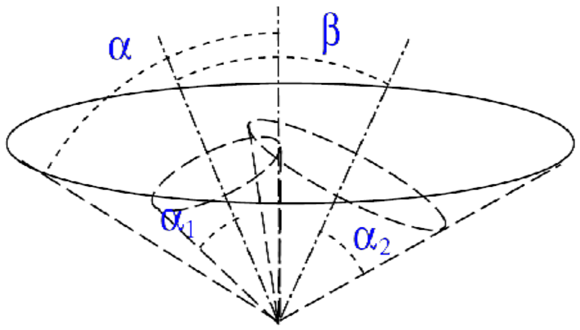


Figure 6. Combination of Two Sub-node Cone Apex Angle

If some node has only one sub-node, its α attribute equals α value of its sub-node. Each leaf node of binary tree only includes one triangle element, i.e., $\alpha = 0$.

The tri-patches shall unify the normal vector, thus during the movement process of fabric, above analysis on α angle can be effective. We unifies the normal vector of all tri-patches when initialization.

Detect some node of hierarchical bounding box tree.

1) If $\alpha < \pi/2$, it can be determined that no self-collision occurs in this area, then stop further detection; 2) If $\alpha \geq \pi/2$, self-collision may occur in this area.

IV. Detection of Distance Between Triangles' Centroids

The basic thinking of space subdivision based collision detection is: Collision occurs between objects adjacent in space only. Inspired by this thinking, we consider that fabric self-collision occurs between triangles adjacent in space and no collision may occur between two triangles which are too far away in space. This paper uses the distance between centroids of two triangles to identify the distance of the two. When this distance is larger than some threshold value, we can determine that the two triangles will not intersect, otherwise, they may intersect. Here we take isosceles right triangle as an example to illustrate this principle as below.

Supposing two triangles, $\Delta A_1B_1C_1$ and $\Delta A_2B_2C_2$ are isosceles right triangles in the same size, with leg L, and distance between centroids D. When two triangles collide, the distance must meet $D \le \frac{2}{3}\sqrt{5L^2}$.

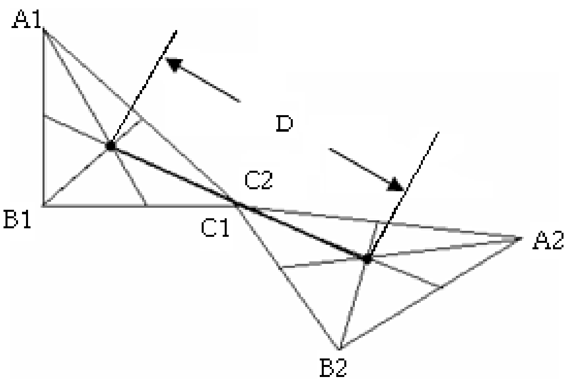


Figure 7. Diagram of Centroidal Distance

When D = $\frac{2}{3}\sqrt{5L^2}$ (as shown in Figure 7), two triangles are coplanar, the centroids of the two triangles and the intersected hypotenuse vertex are on the same line, and it happens that two triangles intersect and the distance between the two's centroids is the farthest.

Based on above principle, this paper gives a detection method using the distance between centroids. We sets a distance threshold value, if the distance between centroids of two triangles is larger than it, the two triangles do not intersect. Thus we filter out this triangle paris and do not perform accurate intersection detection any more. Though above determination method may filter out some collision pairs which may occur in fact, it can greatly improve the efficiency and the loss of a few collision pairs will not fatally impact the reality of simulation, and it may further adjust distance between centroids threshold value during the application process according to the reality demand of system to get the expected reality in an acceptable missing rate.

V. Accurate Detection between Geometric Primitives

After having detected the collision between bounding boxes of two leaf nodes, it is necessary to detect if collision occurs between geometric primitives bounded by two bounding boxes. Point-triangle collision detection is a main composition of this kind of detection. Between two frames, if one particle penetrates the triangle, it can be considered that collision occurs. Then compute the specific location of collision and make right response and avoid penetration.

The traditional point-triangle collision detection decomposes above issue into two processes. Firstly, detect if the particle collides with the plane where triangle is located; if any collision within the plane, detect if this collision point is located in the triangle.

The 1st process is to connect the location at current and next moment of the particle as a straight line, and then determine if this straight line intersects with the plane, as shown in Figure 8. Record the location of particle P at the moment k as Pk and the location of next frame at the moment k+1 as Pk+1.

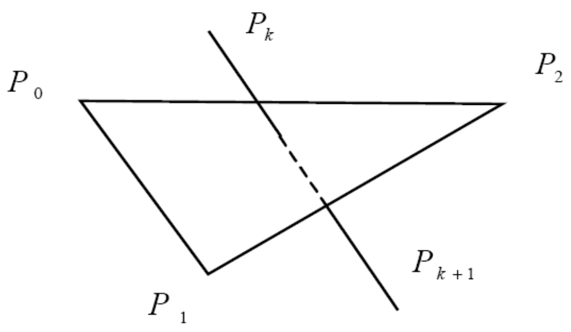


Figure 8. Point-Face Intersection

As the calculation workload of edge-face intersection is rather big, the algorithm is much time-consuming. This paper gives a method reducing the times of edge-face intersection detection, which filters out point-face pairs not needing intersection, thus reduces the calculation workload and improves the real time of fabric simulation. The main thinking of this method: Judge if the location of the particle between two moments is on the same side of triangle. If on the same side, it means the particle does not penetrate triangle, no point-face intersection calculation is needed, thus the optimization goal is achieved. Specific method is as follows:

1）Referring the normal vector and one vertex of triangle, the equation of plane is obtained;

2）Introducing the particle location at current moment and next moment into the equation, x is solved;

x=0 means the point is on the plane; x>0 means the point is on positive side of the plane; x <0 means the point is on the negative side of the plane.

VI. Self-collision Response

Once collision is detected, collision response shall be processed. Response is an important component of collision processing. It involves knowledge like mechanical feedback, movement physics, and other fields [11, 12]. It affects the reality of fabric simulation as well.

When a particle collides with a triangle at certain speed, the speed components along normal vector of triangle will disperse or rebound to some extent (i.e., leave after contact). The collision result is shown as in Figure 9:

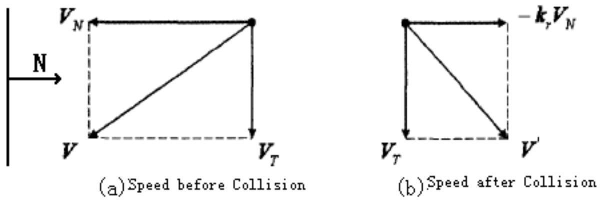


Figure 9. Change in Speed before and after Collision

In Figure 8 (a) V means the speed of particle before collision, (b) V' means the speed of particle after collision. Make an orthogonal decomposition in the direction of normal vector for V, i.e., V_N and V_T as shown in Figure 8 (a). V_T keeps unchanged after collision, while V_N changes in the direction of normal vector.

Firstly, consider the change of V_N when the particle moves in normal vector direction of the triangle. Supposing the anti-collision coefficient of the triangle is k_r ($0 \le k_r \le 1$), no friction factor is considered, $V_N^{'}$ of particle in normal vector direction of the plane after collision turns to $-k_r V_N$, i.e.

$$V_N^{'} = -k_r V_N . \tag{2}$$

Notice that the rebound coefficient k_r is physical property of fabric, so they are different depending on varied fabric performances.

Secondly, consider the change of V_T when the particle moves in normal vector direction of the plane. In the real world, frictional force exists everywhere. In order to improve the reality of simulation, it is necessary to analyse friction force in system. When the particle collides with triangle, the particle rubs with triangle mutually. Friction force damps horizontal movement speed of particle.

Supposing k_f is the friction coefficient, if $\|V_T\| \ge k_f \|V_N\|$, the particle will make friction movement horizontally on the triangle, the horizontal component of speed will attenuate to

$$V_T^{'} = V_T - k_f \|V_N\| U_T \tag{3}$$

$$U_T = V_T / \|V_T\| \tag{4}$$

If $\|V_T\| < k_f \|V_N\|$, the particle will not slide but keep still in horizontal direction, i.e. $V_T^{'} = 0$.

It is not difficult to infer, if $k_f = 0$, the particle will slide in a friction-free way on the surface; if $k_f = \infty$, the particle will not slide on the surface. Friction coefficient k_f is physical property of fabric, which differs depending on different types of fabrics.

Finally, the speed after collision can be obtained with the analysis on two speed components, i.e.

$$V' = V_N^{'} + V_T^{'} . \tag{5}$$

VII. Experimental Result and Analysis

Experimental environment:

OS: Windows XP; CPU: Intel dual-core 2.0GHz; Memory: 2.0GB; Programming environment: VC++ and OpenGL

A. *Experiment 1: Quad Tree*

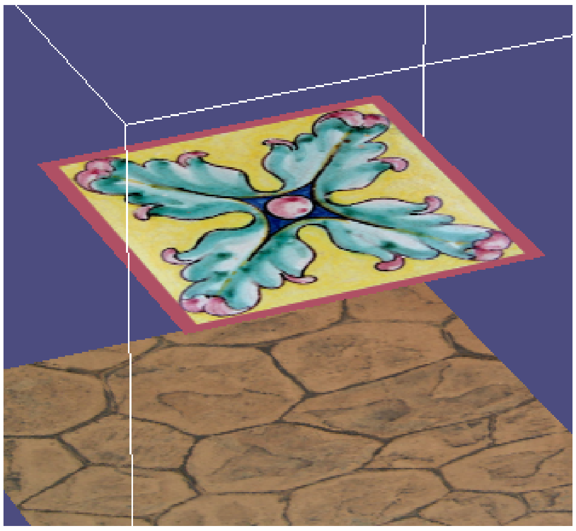
This experiment compares quad tree based collision detection method with traditional binary tree method.

Firstly, make a comparison on the number of bounding boxes, shown in Table I. Compared to binary tree, quad tree reduces the total number of bounding boxes and reduces system calculation consumption accordingly.

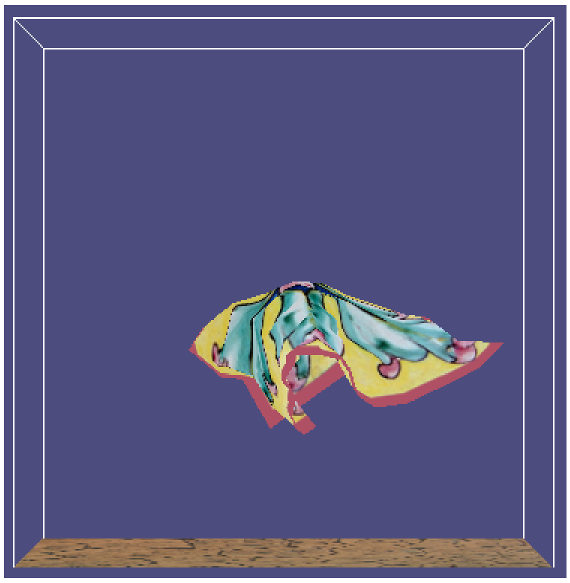
TABLE I. Number of Bounding BoxEX

Number of Particle	5×5	9×9	17×17
Binary Tree	63	255	1023
Quad Tree	53	213	853

Then we do an experiment for the detection efficiency of the two methods. As in Figure 10, we create a fabric in horizontal direction, fix the central part of fabric, and simulate the fabric dropping and the self-collisions of each part during the progress. Count the number of collision pairs (see Table Ⅱ) and average frame rate within 10 seconds (see Table Ⅲ) detected by two methods.



(a)



(b)

(a) Create a Fabric In Horizontal Direction
(b)Drop the Fabric

Figure 10. Simulation of Fabric Dropping

TABLE II. NUMBER OF DETECTED COLLISION PAIRS

Number of Particle	5×5	9×9	17×17
Binary Tree	30488	504154	895692
Quad Tree	21960	186565	352657

TABLE III. COMPARISON OF AVERAGE FRAME RATE OF BOUNDING BOX TREE METHOD (FPS)

Number of Particle	5×5	9×9	17×17
Binary Tree	65.09	56.82	16.65
Quad Tree	68.58	62.20	37.64

According to Table II, III, we can find out that the number of collision pairs needing accurate intersection is apparently reduced by quad tree method, which means quad tree method may filter out more collision pairs not collided and accordingly reduce the times of accurate intersection detection of point-face and edge-edge later.

From Figure 11, we can see that the more particles, the more apparent in improving system efficiency.

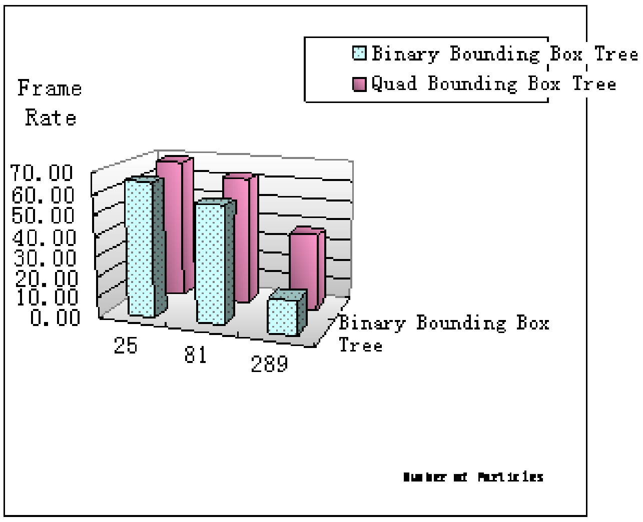


Figure 11. Comparison of Average Frame Rate

B. Experiment 2: Detection Method according to Distance between Tri-Patches' Centroids

This experiment is to verify the effectiveness of detection method according to the distance between tri-patches' centroids.

Create a fabric in horizontal direction, fix the central part of fabric, and simulate the fabric dropping and the self-collisions of each part during the progress. Respectively count the number of collision pairs (see Table IV) and average frame rate when 100000 collision pairs are detected (see Table V).

TABLE IV. DETECTED COLLISION PAIRS

Number of Particle	5×5	9×9	17×17
Unused Distance between Centroids	100000	100000	100000
Detection Method using Distance between Centroids	83193	76022	63606

TABLE V. COMPARISON OF AVERAGE FRAME RATE (FPS)

Number of Particle	5×5	9×9	17×17
Unused Centroidal Distance Detection Method	73.46	71.26	49.07
Centroidal Distance Detection Method	74.90	74.36	56.75

Within detection method using distance between centroids, the more particles, the more filtered collision pairs. And the frame rate is apparently improved as well.

C. *Experiment 3: Parameters of Distance between Centroids*

This experiment verifies the impact of parameter on the efficiency of detection method using distance between triangles' centroids.

Record the longest side length of triangle as L and take centroidal distance parameter respectively 1.5L, 1.4L, 1.3L, 1.2L, 1.1L, L, 0.9L, 0.8L, 0.7L, 0.6L, 0.5L.

Create a fabric with 17*17 particles in horizontal direction, fix the central part of fabric, and simulate the dropping of fabric. Count the filtering rate and missing rate when the total number of collision pairs reaches 100000 under different centroidal distance parameters (See Table VI).

TABLE VI. FILTERING RATE OF COLLISION PAIRS AND MISSING RATE OF COLLISION PAIRS

Parameter	Filtering Rate of Collision Pair	Missing Rate of Collision Pair
0.5L	19.92%	3.10%
0.6L	21.10%	3.93%
0.7L	22.79%	5.25%
0.8L	25.93%	7.42%
0.9L	29.59%	9.89%
1.0L	36.39%	12.91%
1.1L	48.02%	18.76%
1.2L	55.97%	25.72%
1.3L	64.02%	32.23%
1.4L	70.38%	39.96%
1.5L	81.10%	49.21%

Referring to Table VI and Figure 12, detection method using distance between triangles' centroids can improve the system efficiency on the premise of losing a few effective collision pairs; the relation between distance parameter and the missing rate of effective collision detection is: The larger the distance, the bigger the missing rate of collision. The distance parameter can be adjusted according to the system demand to improve frame rate on the premise of acceptable missing rate of collision pair.

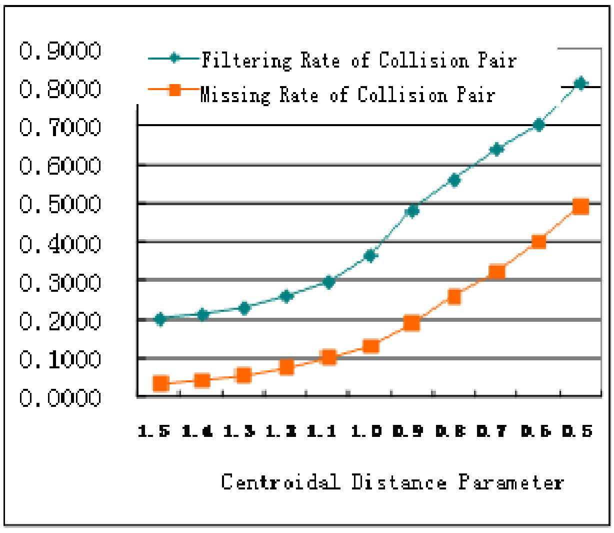


Figure 12. Centroidal Distance Parameter

VIII. CONCLUSIONS

Regarding the character of cloth deformation, this paper gives a set of quad tree based self-collision detection method.

1) A construction method of hierarchical bounding-box tree based on the connectivity of fabric geometric primitives from top to down is given. This method takes the spatial relation of primitives as criteria and merges node from top to down, which provides necessary information for self-collision detection.

2) During detection process, we use a self-collision detection method based on quad tree and AABB bounding-box, which improves the efficiency of collision detection by using normal-vector cone determination rules.

3) A detection method according to distance between triangles' centroids is given. This method can improve calculation efficiency greatly on the premise of missing a few collision pairs.

4) Regarding point-triangle collision detection, our method make a rule that whether a line-plane pair should be continued for intersection detection depends on whether the particle between two adjacent moments is located by the same side of the triangle.

The experiment results show that our methods can effectively improve the efficiency of collision detection on the premise of keeping relatively high realistic simulation effect.

ACKNOWLEDGMENT

The authors wish to thank Yueyuan Yang and Jia Zhao for preview work and useful discussions. Yueyuan Yang did a lot of important work on cloth simulation and deformation. Jia Zhao researched on physical collision detection and response of rigid body. They offered the authors help and inspiration.

This research is supported in part by a grant from the National High-Tech Research and Development Plan of China under Grant No. 2007AA01Z337.

REFERENCES

[1] David Baraff, Andrew Witkin. Large Steps in Cloth Simulation. Computer Graphics, 1998,33(4):43-57

[2] M. Desbrun, P. Schroder and A. Barr. Interactive Animation of Structured Deformable Objects. In Graphics Interface, 1999: 1-8

[3] Doug L. James, Kayvon Fatahalian. Precomputing Interactive Dynamic Deformable Scenes, ACM Transactions on Graphics (ACM SIGGRAPH 2003), 22(3): 879-887

[4] R. Bridson, R. Fedkiw, and J. Anderson. Robust Treatment of Collisions, Contact and Friction for Cloth Animation. ACM Transactions on Graphics (ACM SIGGRAPH 2002), 21(3):594-603

[5] David Harmon, Etienne Vouga, Rasmus Tamstorf, and Eitan Grinspun. Robust Treatment of Simultaneous Collisions, SIGGRAPH (ACM Transactions on Graphics) 2008, 27(3):1-4

[6] Andrew Selle, Jonathan Su, Geoffrey Irving, and Ronald Fedkiw. Robust High-Resolution Cloth Using Parallelism, History-Based Collisions, and Accurate Friction, IEEE Transactions on Visualization and Computer Graphics (TVCG), 2009, 15(2): 339-350

[7] Min Tang, Sean Curtis, Sung-Eui Yoon, and Dinesh Manocha, ICCD: Interactive Continuous Collision Detection between Deformable Models Using Connectivity-Based Culling, IEEE Transactions on Visualization and Computer Graphics, 2009, 15(4):544-557

[8] Naga K. Govindaraju, David Knott, Nitin Jain, Ilknur Kabul, Rasmus Tamstorf, Russell Gayle, Ming C. Lin, Dinesh Manocha. Collision Detection Between Deformable Models Using Chromatic Decomposition. ACM Trans. on Graphics (Proc. of ACM SIGGRAPH). 2005.24, 3, 991–999.

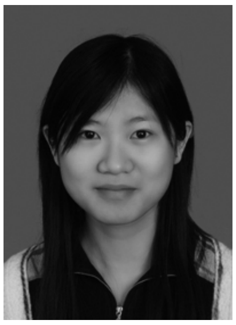
[9] Sean Curtis , Rasmus Tamstorf , Dinesh Manocha, Fast Collision Detection for Deformable Models Using Representative-triangles, Proceedings of the 2008 Symposium on Interactive 3D Graphics and Games, February 15-17, 2008

[10] Provot X., Collision and Self-collision Handling in Cloth Model Dedicated to Design Garment [A], In: Proceeding of Graphics Interface'97, Kelowna B.C., 1997: 177-189

[11] D.Baraff. Interactive simulation of solid rigid bodies. IEEE Computer Graphics and Applications, 1995, 15(3): 63-75

[12] J.M.Snyder. An Interactive Tool for Placing Curved Surfaces without Interpenetration. ACM Computer Graphics, 1995, 29(4):209-218

Bing He born in 1971. He received his BS, MS and PHD degree in electronic engineering from Beihang University. He is an associate professor in computer science department of Beihang University. He is working in the State Key Laboratory of Virtual Reality Technology and Systems. His current research interests include virtual reality, visualization, information fusion, image and video processing, recognition AI.



Liu Cheng received her BS and MS degree in computer science and engineering from Beihang University, where she pursued research in graphic and simulation.