Research of Fast Cloth Simulation Based on Mass-Spring Model

Cao Zhenfang
State Key Lab of Virtual Reality Technology and Systems
Beijing University of Aeronautics and Astronautics
Beijing, China, 15101522311
h.t.rain@qq.com

He Bing
State Key Lab of Virtual Reality Technology and Systems
Beijing University of Aeronautics and Astronautics
Beijing, China, 13911026538
hebing@buaa.edu.cn

***Abstract*—In cloth simulation based on the spring-mass model, algorithm dealing with super-elasticity reduces the simulation efficiency to some extent. Based on the correction algorithm of modifying position and velocity, an improved algorithm is proposed which can improve the efficiency effectively. Meanwhile, for collision detection, a cloth tree updating algorithm is proposed to improve the balance of cloth tree and speed up cloth collision detection efficiency.**

Keywords- Cloth Simulation; Mass-Spring; Super-Elasticity

I. INTRODUCTION

In computer animation field, Terzopoulos[1] is the first to put forward the method of cloth simulation based on physical model. At the SIGGRAPH 1992 Conference, Carignan[2] in his article added damping and self collision based on the physical model raised by Terzopoulos. In the same year, Breen[3] proposed to simulate natural draping effects of cloth and tablecloth based on a discontinuous particle model. Before long, Eberhardt[4], based on Breen's model, raised a simulation method based on Lagrange Formula. In 1998, Volino[5] put forward a novel method based on deformable body theory and Newtonian mechanics. Eischen[6] then raised a method based on nonlinear finite element.

A critical step in cloth simulation animation is to digitalize the physical model. At SIGGRAPH 1998 Conference, Baraff and Witkin[7] proposed a method based on implicit differential equation. This technology improves rendering efficiency and sense of reality. By calculating Hessian matrix in advance to speed up simulation, Desbrun[8] achieved real-time rendering. In the article published in 2000, Kang[9] put forward a method to improve implicit differential equation method so as to enhance sense of reality and real-time performance.

Goldenthal[10] made further study on cloth deformation. Study on fine-scale cloth wrinkles and accurate physical characteristics made by Selle[11] in 2009 significantly improved simulation effect of wrinkles and folds. However, to achieve vivid effect, calculation needs to be made, which are time-consuming. Wang[12] published an article in 2010, in which, he generated a precomputed database built by simulating the high-resolution clothing as the articulated figure is moved over a range of poses. When real-time rendering is made, they drive wrinkle generation from the pose of the figure's kinematic skeleton. This method can combine with a coarse simulation to produce the final results at interactive rates, but its shortage is that it can only simulate tight clothes.

II. RELATED WORKS

A. Super-elasticity Constraint Algorithm

The method of simulating cloth based on spring-mass model is acquired from elastic deformable model. However, cloth also has non-elastic feature, so cloth deformation scale must be taken into consideration so as to make the performance more realistic. When a small part of the cloth surface bears large concentrated force, large spring deformation will cause unnatural stretching of cloth, and local deformation appears less realistic than the real cloth deformation, this is super-elasticity. There are many methods to work on the problem of super-elasticity, for instance, improving spring stiffness coefficient, adjusting particle position based on kinetic equation and modifying particle position and velocity orderly, etc. Traditional ways to settle the problem of super-elasticity include: particle velocity correction, particle position and velocity correction and adoption of non-linear elastic modulus. Particle position correction and velocity correction virtually belong to one category, that is, correct physical parameters of the particle at the end of each frame artificially. In the article published in 2009, Selle[11] adopted a method to settle the problem of super-elasticity, but iterative calculations need to be carried out, and this method is time-consuming because of the large number of cloth model particles.

B. Collision Detection Algorithm

Spatial structure tree of cloth should be constructed at the preprocessing stage before simulation, and it should be updated dynamically in each frame during simulation. There are two tasks to accomplish during the structural tree updating process: the first is to keep balance of the spatial structural tree as much as possible, and the second is to reduce the number of node added/deleted for each frame. Therefore, generating a new node one time or deleting an existing node will lead to allocation and release of the above dynamic memory. An effective solution to solve this problem is to apply more memory areas at the pre-processing stage so as to ensure large amounts of memory operations when updating the structural tree. However, given that the number of triangles relates to complexity of cloth model, so spatial structural tree node quantity estimation should be conducted at the pre-processing

stage of different models each time, thus this strategy is low in efficiency.

A cloth structural tree building and updating method is put forward in this article. This method can keep the structural tree balanced when updating at the end of per frame, and the number of tree nodes will not change during the whole simulation process after a structural tree is built, thus avoiding frequent dynamic memory operations.

III. SUPER-ELASTICITY CONSTRAINT ALGORITHM

A. *Physically-Based Cloth Modeling*

The cloth model used in this article adopts triangle with three particles as basic unit. We assume that particles are connected with linear springs, so that we can calculate force produced by the springs to simulate cloth elasticity. Tensile property of the spring relates to the cloth material and its physical properties.

In mechanical model used in this article, the force exerted by the particle includes external force F_ext and internal force F_int. The external forces mainly include gravity and air resistance, of which, gravity is constant, and damping force changes in proportion to particle's velocity. The internal force refers to applied force among particles, which mainly consists of stretching force, shearing force and bending force. Hence we get mechanics equation as follows:

$$m\frac{\partial^2 X}{\partial t^2} = F_{int}(X,t) + F_{ext}(X,t) \quad (1)$$

B. *Numerical Integration Method*

Runge-Kutta Method is a high-precision and per time step algorithm which is widely applied in engineering. This method has high precision, but it has high computation complexity and is time-consuming compared with explicit Euler integration method. In this article, instead of using these two methods, we carry out Taylor expansion at time (t-Δt) and (t+Δt) which is before and after the simulation moment respectively:

$$x(t+\Delta t)=x(t)+\dot{x}(t)\Delta t+\frac{1}{2}\ddot{x}(t)\Delta t^2+(t)\Delta t+O(\Delta t^4) \quad (2)$$

$$x(t-\Delta t)=x(t)-\dot{x}(t)\Delta t+\frac{1}{2}\ddot{x}(t)\Delta t^2-(t)\Delta t+O(\Delta t^4) \quad (3)$$

Here we add (2) and (3):

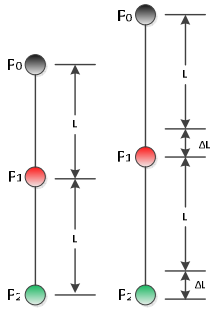
$$x(t+\Delta t)=2x(t)-x(t-\Delta t)+\ddot{x}(t)\Delta t^2+O(\Delta t^4)$$

$$=x(t)+[x(t)-x(t-\Delta t)]++O(\Delta t^4) \quad (4)$$

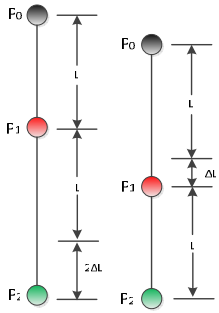
C. *Synthesized Optimized Superelasticity Constraints Algorithm*

There are several ways to deal with the problem of super-elasticity, including particle velocity correction, particle position and velocity correction and adopting non-linear elastic modulus. Selle adopted a method to settle the problem of super-elasticity in 2009, but iterative calculations need to be conducted, and this method is time-consuming because of the large amounts of particles. In this article, we put forward an improved super-elasticity algorithm which has higher efficiency than that of algorithm raised by Selle.

The process of the Selle's algorithm is shown in Fig.1, P0, P1 and P2 are endpoints of the spring, of which, to take the case of fixed point into consideration, we set particle P0 as the fixed point, which means that its velocity and position will not change under the spring force. The particle velocity generally equals to human skeleton velocity. In Fig. 1(a), springs P0P1 and P1P2 are at initial state, distance L refers to balanced distance between them. After simulating for a time step, or when drawing the next frame, we find that these two springs all have super-elasticity problem, as shown in Fig. 1(b).



(a) Initial state (b) Super-elasticity



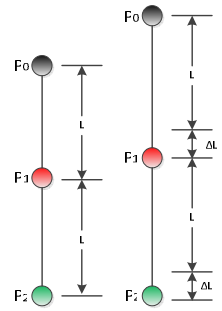
(c) The first correction (d) The second correction

Figure 1. Correction based on velocity and position

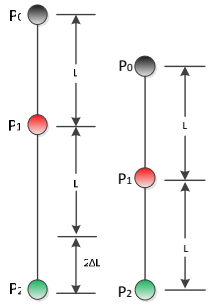
Of which, ΔL>0.1*L. Based on velocity and position correction algorithm for super-elasticity, we then need to correct their positions or velocities. Fig. 1(c) shows correction of spring P0P1's position, as P0's position will not change, we correct particle P1 toward the direction of particle P0, then the length of spring P0P1 gets back to L. Super-elasticity of spring P1P2 becomes more severe as a result of P1 position correction, and length of spring P1P2 exceeds equilibrium state length 2ΔL. When correcting position of spring P1P2, as particles P1 and P2 are non-fixed particles, their positions need to be corrected, as shown in Fig. 1(d). After spring P1P2 is corrected, its length resumes to equilibrium state, while correction of particle P1 position leads to super-elasticity of spring P0P1, which exceeds equilibrium state length ΔL. From Fig. 1 (b) and (d), we can see that, spring P0P1 still has super-elasticity after carrying out

position correction algorithm, that means there are half super-elasticity remain unsettled. Therefore, iterative computations on particle velocity and position correction algorithm need to be carried out for several times to settle super-elasticity problem of all springs.

Since we have to traverse all springs at every iteration, we could save much time if we can make one less iteration. This article puts forward super-elasticity algorithm with the aim to reduce iteration computation times and improve simulation algorithm efficiency. To begin with, each spring particle is marked with a priority level. Once the spring has super-elasticity and been corrected once, the priority level of the spring endpoint will be one level lower. If one of the two particles of a spring has higher priority level, correct the position of the particle with higher priority. If the two particles have the same priority level, then we need to calculate relative velocity of the particles. If the relative velocity of particle is larger than 0 in the spring increase direction, correct position of this particle; if the relevant velocity of particle is 0, correct the two particles simultaneously; if the relevant velocity of particle is less than 0, correct position of another particle. Move P0P1 spring first, then lower priority of P1 by one level, and then correct spring P1P2. As priority of P1 is smaller, correct position of P2. From Fig. 2(b) and 2(c), we can see that the two springs after correction have no phenomenon of super-elasticity, so the correction result is more accurate than that by adopting Selle's method.



(a) Initial state (b) Super-elasticity



(c) The first correction (d) The second correction

Figure 2. Algorithm based on particle velocity and position

IV. CLOTH COLLISION HANDING

A. *Collision Detection and Response Process*

In this article, unified algorithm process is adopted to handle with collision between clothing and human body, as well as clothing self-collision.

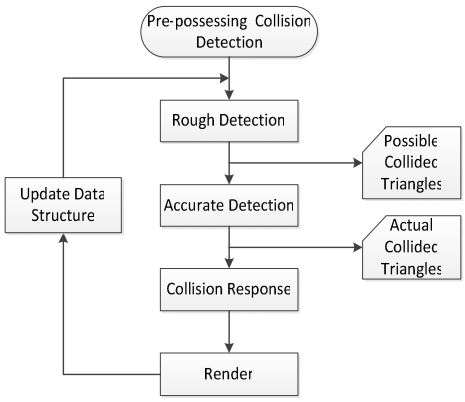


Figure 3. Collision algorithm flowchart

First, at the pre-processing stage of collision detection, construct proper data structure for the object in the scene, or generate bounding box according to triangular mesh of the object, then establish structural tree according to their spatial position relationship. Binary tree structure of AABB bounding box is adopted in this article.

Then, we enter into collision detection stage, which includes rough collision detection and accurate collision detection:

At the rough collision detection stage, we use bounding box intersection test technique as well as binary tree structure to carry out top-down bounding box intersection test. After areas which are impossible to be intersected are removed, we get the bounding box pair which might collide.

At the accurate collision detection stage, we conduct precise intersection test on basic geometric primitive contained in each pair of triangles, for instance, point-triangle pair and edge-edge pair, etc. If the geometric primitive pairs have collided, we will record the primitive pair and store relevant collision information for future use at collision handling stage.

At the collision response stage, we calculate collision response kinetic equation based on collided geometric primitive pair and correct position and velocity of collision vertex while avoiding penetration among models. After getting final position and velocity of each particle, we render the full scene.

B. *Rapid Collision Structural Tree*

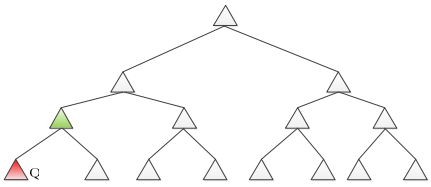
In this article, we put forward a new way to build and update cloth structural tree in response to the above mentioned issues, this method can better maintain balance of the structural tree when updating at each frame, and the number of tree node will not change during the whole simulation process after structural tree construction, thus avoiding frequent dynamic memory operations.

The tree structure shown in Fig. 4 is cloth structural tree, and each node of the structural tree represents a triangle, cloth bounding box is stored in leaf node of the structural tree, of which, red leaf node means node to be updated (node Q). Update leaf node Q as follows:

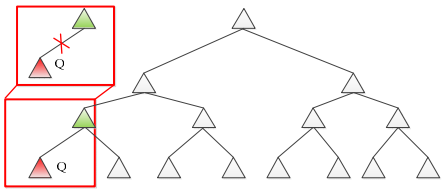
(1) Cut off all edges connecting node Q with its parent node, and update all parent bounding boxes of the node Q, see Fig. 4(b);

(2) Carry out depth-first traversal starting from root node of the structural tree. Each time we traverse a node, we directly calculate the distance between the updated node bounding box and the two child nodes bounding box of the traversed node, instead of judging whether node bounding box contains that of the updated node during traversal, then we carry out depth-first traversal of close child nodes up to the leaf node. As shown in Fig. 4(c), red triangles Q1, Q2, Q3 and Q4 are depth-first traversal path. Fig. 4(c) insert updated node into the structural tree.

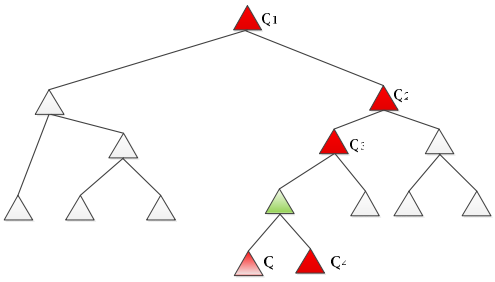
During this node updating process, the total nodes number of the whole structural tree remains unchanged, and the time complexity of the traversal is O(logN). No matter updated node is inserted to which layer of the structural tree, the tree nodes number will remain the same during the updating process.



(a) Initial state of the structural tree



(b) Update structural tree node



(c) Result of the updated structural tree

Figure 4. Updating process of cloth structural tree nodes

The structural tree updating algorithm described in this article not only avoids prediction targeting at cloth bounding box quantity, but also has good balance. Selle adopted bottom-up structural tree construction algorithm, when nodes are updated, traversal is not made up to leaf node; instead, find the first node which do not contain updated node bounding box, then insert updated node at this layer. As shown in Fig. 5, as a result of updated node insertion, that is to insert green node into the second layer of the structural tree, all child nodes' depth value of the green node increased, the higher the depth value of the tree, the poorer balance is. Therefore, Selle's method cannot effectively reduce depth value increases of the structural tree. The structural tree updating algorithm described in this article requires that we must search leaf node before conducting insertion operation during node updating each time. From Fig. 4 (c) we can see that, our method can prevent updated node from being inserted at higher level of the tree which leads to the child nodes' depth increasing. So at the collision detection stage search path for our method is shorter, and the efficiency is higher.

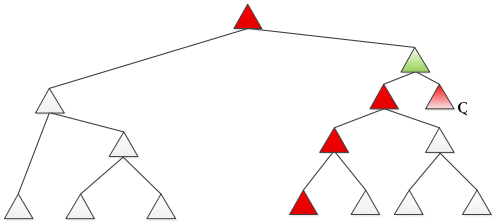


Figure 5. Selle's structural tree updating algorithm

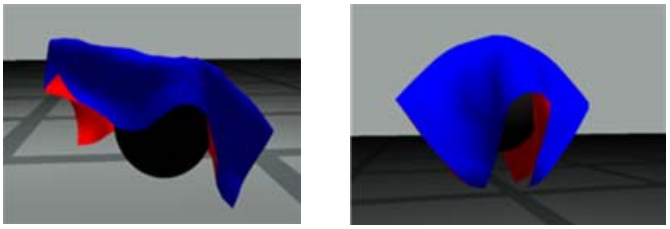
V. EXPERIMENT RESULTS

A. *Experiment I: Cloth Collision Handling Effect*

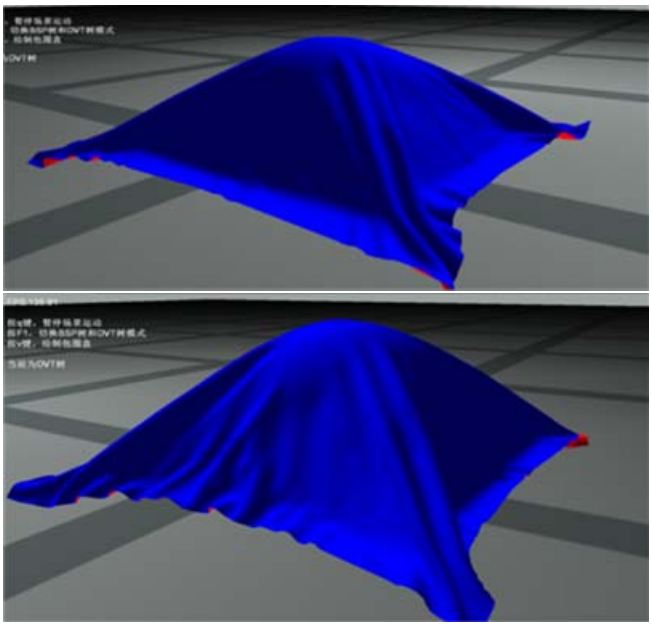
Experiment I simulates a scene that a cloth falls on a sphere, see Fig. 6(a). If the particle quantity is small (10*10), the rendering effect of cloth model is stiff. As shown in Fig. 6(b), when the particle quantity reaches 10000(100*100), the simulation effect is relatively realistic, and its simulation frame rate is about 2fps when cloth self-collision case is not taken into account.

B. *Experiment II: Structural Tree Efficiency Comparison Experiment*

In this experiment, we compare efficiency of bottom-up structural tree method after improvement and Selle's top-down structural tree method. The vertical ordinate in Fig. 7 represents time for updating a structural tree once, and the unit is millisecond. The horizontal ordinate represents particle numbers of fabric. When the particle quantity of cloth is small, the efficiency of the algorithm adopted in this article is slightly lower than that of Selle's algorithm; when mass quantity increases, the efficiency of the algorithm adopted in this article begin to be higher than Selle's method. The experiment result suggests that, when the mass quantity of fabric is small, balanced structural tree decreases simulation efficiency; when the mass quantity of fabric is large, the improved bottom-up structural tree method has relatively high efficiency, thus the structural tree has good balance performance.



(a) 10*10 Particle cloth and sphere collision effect



(b) 100*100 Particle cloth and sphere collision effect

Figure 6. Cloth with different particle quantities and sphere collision effect

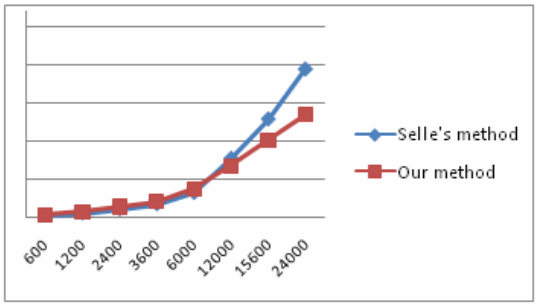


Figure 7. Structural tree algorithm efficiency comparison

VI. CONCLUSION

In this article, we put forward an improved collision handling method, which adopts bottom-top structural tree, and when the structural tree is updated, it can better ensure balance performance, and reduce quantity of node added/deleted. Experimental result indicates that, based on the fabric model used in this article, bottom-up structural method has higher simulation efficiency than the top-down structural method. Compared with Selle's bottom-up structural method, when the particle quantity of the fabric model is less, the method has lower efficiency than that of Selle's method, but when the particle quantity of the fabric model is larger, the method has better balance performance. It can effectively reduce searching time of collision detection, and has higher simulation efficiency.

ACKNOWLEDGMENT

This work was supported by grant No. 61272346 from NSFC (National Natural Science Foundation of China).

REFERENCES

[1] Demetri T., John P. Alan B. and et al, “Elastically deformable models,” Proceedings of the 14th annual conference on computer graphics and interactive techniques, pp. 205-214, 1987.

[2] Michel C., Ying Y., MagnenatT. and et al, “Dressing animated synthetic actors with complexdeformable clothes,” In Proc. of ACM SIGGRAPH, vol (26), pp. 99-104, 1992.

[3] Breen D. E., House D. H. and Wozny M.J., “Predicting the drape of woven cloth using interacting particles,” Proceedings of SIGGRAPH, pp. 365-372, 1992.

[4] Bernhard E., Andreas W. and Wolfgang S., “A fast, flexible, particle-system model for cloth draping,” IEEE Computer Graphics and Applications, vol(6), pp. 2-59, 1996.

[5] Pascal V., Martin C. and MagnenatT., “Versatile and efficient techniques for simulating cloth and other deformable objects,” Proceedings of SIGGRAPH, pp. 137-144, 1992.

[6] Eischen J.W., Shigan D. and Clapp T.G., “Finite-element modeling and control of flexible fabric parts,” IEEE Computer Graphics and Applications, vol(16), pp. 71-80, 1996.

[7] David B. and Andrew W., “Large steps in cloth simulation,” Proceedings of SIGGRAPH, pp. 43-54, 1998.

[8] Mathieu D., Peter S. and Alan B., “Interactive animation of structured deformable objects,” In Proceedings of the 1999 conference on Graphics interface '99, pp. 1-8, 1999.

[9] Young-Min K., Jeong-Hyeon C., Hwan-Gue C. and et al, “Fast and stable animation of cloth with an approximated implicit method,” Computer Graphics International, pp. 247-256, 2000,

[10] Goldenthal R., Harmon D., Fattal R. and et al, “Efficient Simulation of Inextensible Cloth,” In Proc. of ACM SIGGRAPH, 2007.

[11] Selle A., Su J., Irving G. and et al, “Robust highresolution cloth using parallelism, history-based collisions, and accurate friction,” IEEE Transactions on Visualization and Computer Graphics, pp. 339-350, 2009.

[12] Huamin W., Florian H., Ravi R. and et al, "Example-Based Wrinke Synthesis for Clothing Animation,” In Proc. of ACM SIGGRAPH, vol(29),2010.