Open Access

A New Self-Collision Detection Method for Cloth Simulation

Fengquan Zhang*, Jianfei Wan and Jiaojiao Guo

College of Computer Science, North China University of Technology, Beijing, 100141, China

Abstract: In the system of cloth simulation, the process of cloth deformation and the method of self-collision detection have great significance for fast cloth deformation and realistic rendering. In this paper, we first presented a novel contour test based on the star-shaped property of a polygon, *i.e.* the central data structure for efficient execution of the star-contour test. Then, the paper describes and compares in detail, directly accumulation method and global stiffness matrix method. Taking efficiency, stability and convergence into consideration, the paper puts forward an explicit-implicit combined method which is suitable for force models. At last, a fabric deformation constraint method has been designed, which is used for linear FEM. This method is effective to restrain the fabric from excessive deformation behavior and create realistic simulation results.

Keywords: Star-contour, Self-collision detection, Cloth deformation, Finite Element Method (FEM), Numerical integration.

1. INTRODUCTION

In the simulation of flexible body, cloth not only collides with surrounding objects but also undergoes self-collision. If not appropriately responded, self-collision would manifest as penetration on the cloth, significantly reducing realistic feature. The detection and responding are both demanding and will shed immediate impact on simulation speed.

Most algorithms available now enhance collision-detection efficiency by reducing the number of detection elements. The detection can be handled by extraction of hierarchical structure. This type of algorithms lose their applicability in self-collision since no collided grid can be found between adjacent triangular mesh in independent test. In this paper, we present screening criteria based on regular patch described by direction and contour. Its strength in efficiency and accuracy is then illustrated by experiments.

Cloth simulation categorizes into geometric, physical and a combination of both since 1980s [1, 2]. Geometric method is easy to apply and known for its speed. However, it's relatively bad for simulation method and that's when physical simulation comes in. Mass-Spring [3] and Finite Element Method (FEM) are among the most popular ones. Mass-Spring is extensively used since the algorithm is easy to realize with high efficiency. FEM computes cloth deformation based on its continuous medium and presents more accurate simulation.

2. COMPUTATION WORK FLOW

The proposed self-collision detection algorithm can be widely applied on grid model deformation and cloth simulation. When the grid model is deformed, the algorithm can find the collided triangle patch pair. Whereas in cloth simulation, the algorithm can compute response of the collision node, change the position and speed of collision point, and then realize the wanted collision. Fig. (**1**) illustrates the work flow of the algorithm, consisting of self-detection test on star-contour grid model and cloth simulation.

The method can then be further divided into pre-processing and real-time phases. In the pre-processing, recursive meshing is implemented to build a BVH (Bounding Volume Hierarchy) [4] tree and a SCTT (self-collision test tree). The latter is then traversed in real-time phase. Patch contour of each knot on the two trees will eventually be computed. In the pre-processing phase of cloth simulation, cloth model is first generated, and simulation scene is chosen, then elastic force and numerical integration are computed. Simulation parameters are eventually initialized in real-time phase.

In the real-time phase of self-collision detection for grid model, deformation is achieved by a few known deformation algorithms. The SCTT tree is then traversed to realize the detection, including four steps: a) Patch normal test: one public normal vector is computed for whole patch; b) Star-contour test: to check whether patch projection on the public normal vector intersects with each other; c) BVH test: to check whether one pair patch collides with each other; d) Primitive test: to check whether two triangles collide with each other;

The real-time running in cloth simulation is also divided into five steps:

1) We established a mechanical model for the simulated cloth. By doing so, we can compute external force on all nodes and internal force according to cloth deformation. If the integration method used in the next step is the explicit integration, the force vector can be calculated directly. Otherwise, implicit integration is used for expression of velocity and location.

*Address correspondence to this author at the College of Computer Science, North China University of Technology, Beijing, 100141, China;
Tel: 15910629882; E-mail: fqzhang@ncut.edu.cn

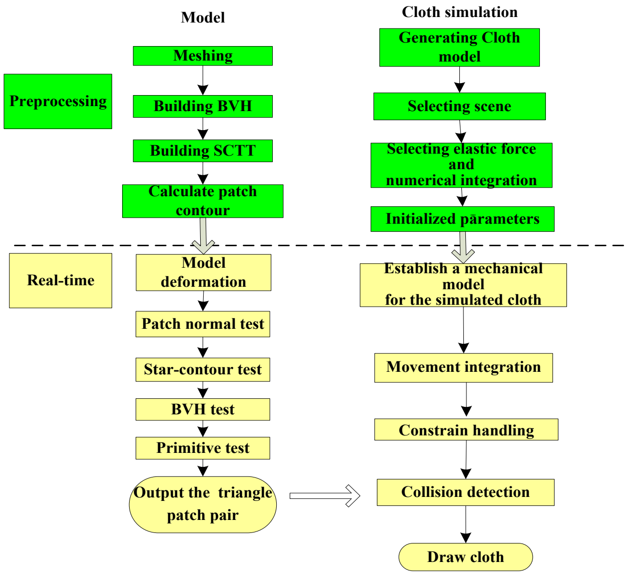


Fig. (1). Self-collision detection & simulation work flow.

2) We established a partial differential equation for force model, which is then solved by numerical integration method. We got the movement of the fabric node status of each frame, including the velocity and location. Hence the dynamic cloth simulation is accomplished. Numerical integration methods applied may be explicit or implicit depending on the force model applied;

3) Based on the location in Step 2, we performed constraint handling to avoid excessive deformation;

4) Based on the velocity in Step 2, we performed continuous collision test and computed collision response of the collided nodes, and changed their position and velocity;

5) We rendered a real-time cloth image based on the changed location information and used the current location and velocity into the simulation for the next frame.

3. KEY TECHNIQUES

3.1. Self-Collision Detection Algorithm Based on Star-Contours

As is described in [1], in collision detection, BVTT tree expresses hierarchical traverse sequence. One tree node that includes two BVH patches performs collision detection. CSTT including two adjacent patches is defined as sub-tree of BVTT tree. SCTT node is adjacent patch pair to perform collision detection.

First, we performed a patch normal test [5] to find a public normal vector for BVTT node contour. Then all sides on contour can have a projection on the plane corresponding to the public normal vector, which requires vector dot product of all triangular patch within BVTT node greater than 0.

Fourteen directional vectors are chosen by image space sampling and then tested for triangular patches to locate the sample passing all tests. If no sample can meet the requirement, then the directional test fails. These fourteen vectors are chosen from the center of BVTT to its eight vertexes and center of six surfaces.

After directional test, BVTT node will enter into star-contour test. There are two conditions to define a star-contour: (1) there has to be one point on internal plane consisting of all sides; (2) any ray from the point intersects with the side on one point. Polygon can be defined as star on the condition that (1), *t*, the lowest intersection point of top-edge ray and *L* is higher than *b*, the lowest intersection point of bottom-edge ray and *L*; (2), searching line intersects with only one top edge. Fig. (**2**) illustrates realization flow of star-contour self-collision detection algorithm.

1) Read the model information of cloth;

2) We performed meshing and build BVH tree, with each node including a set number of triangular patches;

3) SCTT tree is built based on BVH tree. Each node on the SCTT tree has a pair of node of BVH for self-collision detection;

4) Contour information on each node of BVH tree is computed recursively;

5) After the above pre-processing stage, we performed cycle for collision detection on the movement model;

6) Vertexes are updated;

7) SCTT tree is recursively traversed. First, the directional test is performed to see whether there is a community normal between meshes. If no, a BV test follows;

8) If the directional test succeeds, conditions are examined for star-contour. Success in star-contour test means no collision and failure incurs BV test.

Our optimized star-contour algorithm provides contour fragment projection of SCTT node A, and one boundary on top and bottom respectively. The test is done as follows:

1) To compute AABB bounding box for A's contour fragment projection, which means maximum and minimum *x* and *y* coordinates of projection vertexes respectively;

2) To define mid-perpendicular and mid-horizon of AABB bounding box as searching line, first judge the mid-perpendicular L;

3) Intersection point of top segment bottom line is defined as *t*, and intersection point of bottom segment top line is defined as b;

4) If $b>t$, patch contour may intersect with itself and the contour test fails;

5) Otherwise, when $b<t$, by comparing horizontal section between L and top segment, we compute the number of top segments intersecting with L;

6) If there is one intersection point between L and top segment, and this segment has no bottom segment, it means that this patch contour has no intersection within itself and contour test succeeds;

7) Otherwise, since segment *s* may be a loop and intersect with L on more than one sides, the patch contour may intersect with itself.

8) If the test on mid-perpendicular Axis-Aligned Bounding Box fails, another test is due on mid-horizon. Redo steps 3 to 7.

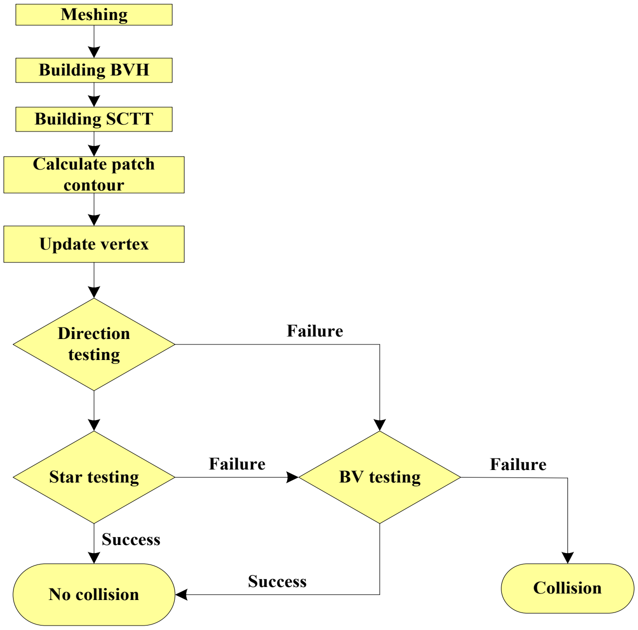


Fig. (2). Star-contour self-collision detection algorithm.

In this way, most elements with no collision are excluded before the test. Collision-detection test efficiency is increased and system real-time requirement is met accordingly.

3.2. Simulation Based on FEM

Considering the fact that soft cloth is subject to deformation, we choose FEM as our mechanical simulation model, for cloth deformation elasticity calculation, numerical integration on dynamic simulation, and constraining method on cloth deformation.

Soft fabric texture deforms caused a change of relative location of node structure. To respond to this specific feature, we established a physical model. Elastic force computing method based on FEM is applied to analyze internal stress such as elastic force and bending force.

FEM Modeling: Finite Element Method (FEM) is an engineering approach which discretizes a grid structure and solves the discrete model by numerical analysis [6, 7]. The cloth model in use is a finite three-node triangular element model, which defines the simulation object as a continuous medium. FEM discretizes this continuous medium into several elements, which are connected by nodes and continue on the boundary. As is shown in Fig. (**3**), our finite three-node triangular model defines cloth as a continuity consisting of several continuous finite elements on the boundary. Elements are in the shape of triangle and the three vertexes of the triangle are nodes of the element. In our model, meshing is in accordance with grid patch meshing to simplify calculation.

Detailed modeling process is elaborated as follows:

First, cloth grid topological structure is established. Each vertex is numbered and assigned a data storage structure consisting of location, speed, weight, *etc*. Each patch is numbered and assigned a data storage structure consisting of index number of its three vertexes in the vertex matrix. Two single-dimensional matrixes store information on vertex and patch respectively.

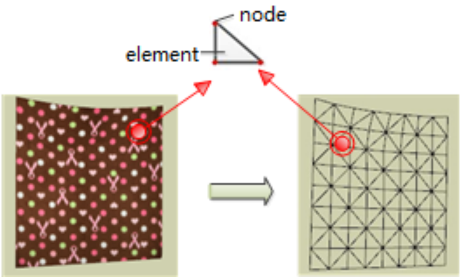


Fig. (3). Cloth FEM.

Second, FEM structure is established. For each element, a data storage structure is established, combining its physical information (elasticity modulus, Poisson's coefficient, etc. to compute elastic force). Here, we also use matrix to store all elements. The element is numbered according to its corresponding patch number in patch matrix.

Cloth deformation under stress: As is described above, stress on the element is supposed to be equivalently integrated on its nodes, which are grid vertexes in our FEM model. In this sense, force analysis is to compute force on all grid vertexes.

Force on each cloth grid includes external and internal force, which can be expressed as:

$$F(i) = F_{external}(i) + F_{internal}(i) \qquad (1)$$

$F(i)$ is resultant force of vertex i, with $F_{external}(i)$ and $F_{internal}(i)$ as its external and internal forces respectively. External force can further be categorized into gravity and air damping force. Gravity is a known constant and damping force is in direct proportion with node velocity. Internal force is acting force caused by internal deformation, and usually consists of elastic force, viscous force and bending force.

FEM based cloth elastic force computing: Elastic force is computed by FEM model as is shown in Fig. (**4**).

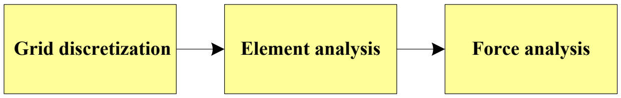


Fig. (4). Work flow of elastic force computed by FEM.

First step, the continuous structure is discretized into several element sets. Elements are connected by nodes;

Second step, acting force on each element is analyzed. Node displacement represents element's deformation. Conversion formula is computed for the elastic force on the nodes;

Third step, resultant force is computed as force analysis.

The primary task for element analysis is to derive conversion relation between node displacement and node force, based on material property, shape, size, number of nodes, position, *etc*. Geometric equations, physical equations, and visual work equations in elastic mechanics are introduced to derive element stiffness matrix, and build an equation on force and displacement. Refer to Fig. (**5**).

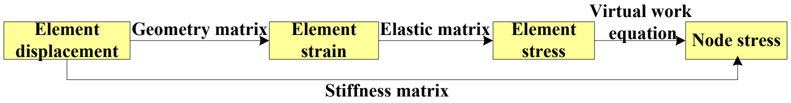


Fig. (5). Element analysis.

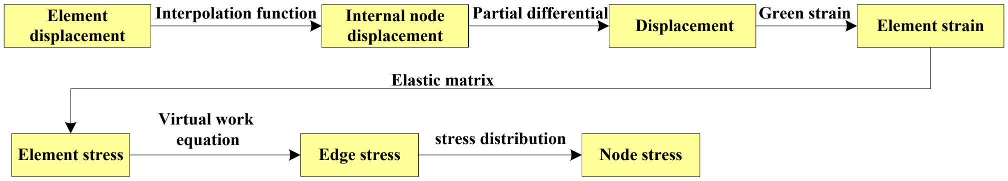


Fig. (6). Element force analysis by green strain method.

After force analysis on single element, we proceed to integral node force analysis. In this process, all forces on the node are integrated to compute the resultant force. There are two approaches applied and compared.

The first method to get the resultant elastic force is direct accumulation of elastic forces on adjacent nodes [8]. The computing is done on the level of element and can further be categorized into Green strain method and Cauchy strain method. The process is illustrated in Fig. (**6**).

Another one takes the force accumulation process as an integration of global stiffness matrix. To put it another way, element force analysis is done first to get element stiffness matrix, which are then integrated into a global stiffness matrix. The global matrix can then generate elastic resultant force of all nodes. The integration of global stiffness matrix can be seen as the force accumulation process.

After building mechanical models for cloth deformation, we apply explicit or implicit numerical integration method to do movement integration and work out movement information on each frame. In response to specific features of our mechanical models and realize cloth dynamic simulation, efficient, accurate and stable integration method is in need. Besides, we put forward a fabric deformation constraint to avoid super-elasticity in simulation.

Numerical integration and deformation constraint: Cloth simulation based on physical models, though with different solving methods, can be summarized into the following process: to express (in force or energy) node elastic relations on account of Newton's laws of motion (or Lagrange motion equation) as one partial differential equation, and then solve the equation by explicit or implicit numerical methods.

In choosing the right numerical method, we take into consideration the system scale, solving accuracy and simulation features. The mechanical model chosen has three features: 1, gravity is a constant, irrelevant to position and velocity; 2, bending force, support force on the collision surface, friction force can only be calculated explicitly (use position and velocity of a known previous step size to compute force directly); 3, elastic force and viscous force can be implicitly expressed in equation, or can be computed directly. The approach we take is a combination of both explicit and implicit integration. The whole process can be explained as follows:

First step, to establish a Lagrange motion equation:

$$M\ddot{x}+C\dot{x}+(K_{g1}x-K_{g2}x_0)=f_{ext} \qquad (2)$$

Second step, to rewrite the above Lagrange equation into the form of Newton's laws of motion:

$$\dot{x}=v \qquad (3)$$

$$M\dot{v}=+Cv-(K_{g1}x-K_{g2}x_0)=f_{ext} \qquad (4)$$

Third step, to further rewrite the equation into the form of integration discrete step sizes:

$$x^{i+1}=x^{i}+\Delta t v^{i+1} \qquad (5)$$

$$Mv^{i+1}=Mv^{i}+\Delta t(-Cv^{i+1}-K_{g1}x^{i+1}+K_{g2}x_0+f_{ext}^{i}) \qquad (6)$$

Fourth step, in order to solve V^{i+1} at $t+\mathrm{V}t$, we put (5) into (6), then:

$$(M+K_{g1}\Delta t^2+C\Delta t)v^{i+1}=Mv^{i}+(f_{ext}-k_{g1}x^{i}+k_{g2}x_0)\Delta t \qquad (7)$$

Last step, we apply conjugate gradient method to solve the equation to have V^{i+1}. Then x^{i+1}, new velocity and location for the node is computed by (5).

As is explained above, the method we use is linear FEM [9], in which the element stress is computed by (8). *p* is Poisson ratio, representing mutual effect of horizontal and vertical deformation. The greater the *p*, the larger the effect is. E is elastic modulus. It can be inferred that a larger *E* means greater elasticity.

$$stress=\frac{E}{1-P^2}\begin{bmatrix}1 & P & 0\\ P & 1 & 0\\ 0 & 0 & \frac{1-P}{2}\end{bmatrix}strain \qquad (8)$$

Instead, if *E* is not a constant and change in accordance with the stress, a non-liner feature can then be simulated. We propose the method to compute element Strain, which is the basis to change *E* by segments.

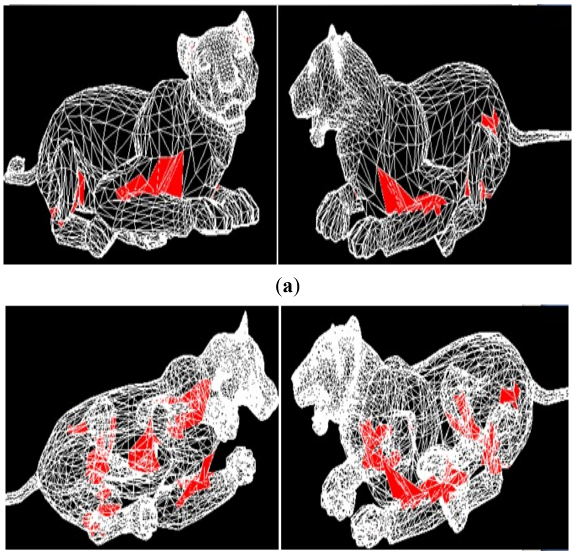
4. EXPERIMENTAL RESULTS AND ANALYSIS

Experimental environment applied in this paper is Windows XP operating system with Intel(R) Core(TM) 4*E7300CPU(2.66GHZ). Development and realization are all in Visual studio 2008. Self-collision detection uses

METIS third-party development library to meshing recursively. OpenGL graphic development libraries are introduced to render scenes and models whereas for cloth simulation, DirectX is used.

4.1. Test results of Star-Contour Collision Algorithm

Animal models are chosen to highlight self-collision in detail. Data recorded are summarized in Table **1**, including time delay, frame rate, number of collision pairs, number of collision pairs after screening, screening rate, *etc*. Screening rate means that the algorithm has already pruned most self-collision. Collision effect is shown in Fig. (**7**), grid patch collision is highlighted with red cells. (**a**) and (**b**) represent non-blank grid and blank grid effect respectively.



(**a**)

(**b**)

Fig. (7). Self-collision in animal grid model.

4.2. Numeric Methods Taken for Elastic Forces

Fig. (**8**) simulates the process of one piece of cloth falling onto a static ball. The number of cloth patch is 512. We use explicit integration and simulation step size is defined as 0.001. Cloth form turns up due to different integration method taken. It is obvious that cloth simulated by that global stiffness matrix shows a well-balanced coverage on the ball whereas cloth simulated by direct accumulation shows one sunken angle and unsymmetrical angles. The former clearly outweigh the latter.

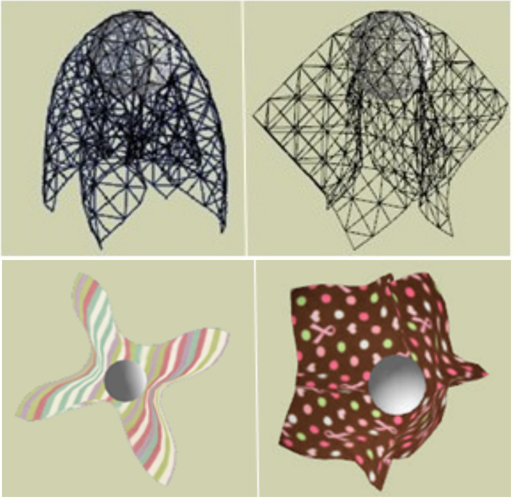


Fig. (8). cloth falling onto a ball: comparison of two methods. left is global stiffness matrix and right is the direct accumulation.

4.3. Comparison Between Two Integration Methods

We also simulate one piece of cloth hanging under gravity with two points fixed. Frame rates simulated by explicit and implicit integration with different numbers grid vertexes (elements) are compared and summarized in Table **2**. Same mechanical method (global stiffness matrix) is applied to rule out other factors.

4.4. System Results

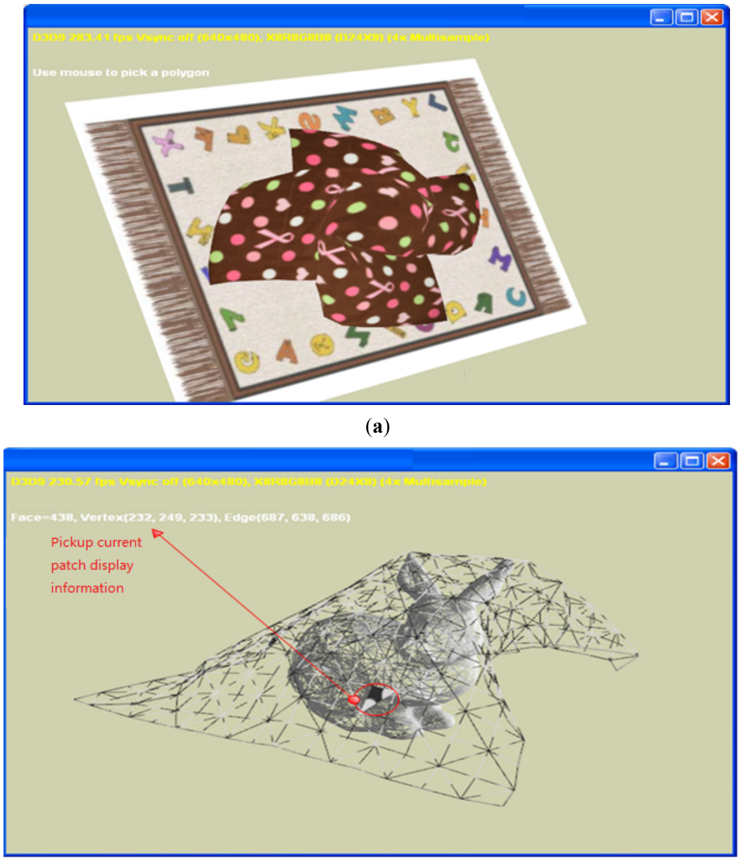
There are two simulations done for final illustration in Fig. (**9**). (**a**) cloth collides with a rolling ball, (**b**) cloth collides with an object in irregular shape. The number of patches is 512 with simulation step size as 0.002.

Table 1. Star-contour algorithm test result.

Model	Bar	Duck1	Duck2	Lion	Horse	Man
Triangles	208	4990	4990	9996	16152	26668
Time delay(ms)	2.576	47.568	49.528	127.482	118.604	163.343
Frame rate(fps)	388	21	20	7	8	6
Number of collision pairs	3608	89358	89358	179866	293987	390954
Number of collision pairs after filtration	958	16145	16406	40507	31874	40254
Filtration rate	73.45%	81.93%	81.64%	77.48%	89.16%	89.70%

Table 2. Average frame rate comparison (Frame/S).

Vertexes	Explicit Method	Implicit Method
9*9	144.75	90.07
11*11	88.05	48.31
13*13	52.53	27.79
15*15	37.55	17.50
17*17	28.09	16.95



D3D9 283.41 fps Vsync off (640x480), X8R8G8B8 (D24X8) (4x Multisample)

Use mouse to pick a polygon

(**a**)

D3D9 230.57 fps Vsync off (640x480), X8R8G8B8 (D24X8) (4x Multisample)

Face=438, Vertex(232, 249, 233), Edge(687, 638, 686)

Pickup current
patch display
information

(**b**)

Fig. (9). Cloth self-collision simulation. (**a**) is cloth-rolling ball collision simulation and (**b**) is cloth-object in irregular shape collision simulation.

CONCLUSION

We studied FEM-based mechanical model simulation and star-contour collision self-detection. More specifically, elastic force computing methods, numeric integration on dynamic simulation, and constraint on cloth deformation are explored. Experiments results indicate that 1) general stiffness matrix shows simulation more close to reality; 2) explicit and implicit combined integration is more suitable for the mechanical model chosen; 3) non-linear feature simulation can effectively constrain cloth over-deformation.

CONFLICT OF INTEREST

The authors confirm that this article content has no conflict of interest.

ACKNOWLEDGEMENTS

This paper is supported by Beijing Natural Science Foundation (4154067), (14YJCZH200), (61402016), (KM20160009008) and (XN071013).

REFERENCES

[1] C.S. Sara, G.P. Alvaro, and A.O. Miguel, "Star-Contours for Efficient Hierarchical Self-Collision Detection", *J. ACM Trans. Graph.,* vol. 29, no. 4, pp. 1-8, 2010.

[2] C.K. Jin, and K.H. Seok, "May Research problems in clothing simulation", *J. Comput. Aided Design,* vol. 37, no. 6, pp. 585-592, 2005.

[3] R. Bridson, R. Fedkiw, and Anderson, "Robust treatment of collisions contact and friction for cloth animation," *J. ACM Trans. Graph.,* vol. 21, no. 3, pp. 594-603, 2002.

[4] S. Curtis, R. Tamstorf, and D. Manocha, “Fast collision detection for deformable models using representative-triangles,” *Proc. Symp. Interactive 3D Graph. Games,* pp. 61-69, 2008.

[5] V. Pascal, and M.T. Nadia, “Efficient self-collision detection on smoothly discretized surface animations using geometrical shape regularity,” *Comput. Graph. Forum,* vol. 13, no. 3, pp. 155-166, 1994.

[6] J.W. Eischen, S. Deng, and T.G. Clapp, “Finite-Element Modeling and Control of Flexible Fabric Parts,” *J. IEEE Comput. Graph. Appl.*, vol. 16, no. 5, pp. 71-80, 1996.

[7] O. Etzmus, M. Keckeisen, and W. Straser, “A Fast Finite Element Solution for Cloth Modeling,” *Proc. Pacific Conference on Computer Graphics and Applications,* Washington: IEEE Comput. Soc., pp. 244-251, 2003.

[8] M. Muller, “The Finite Element Method,” *Real time Physics Class Notes in SIGGRAPH*, Cornell University Zurich, pp. 21-34, 2008.

[9] R. Goldenthal, D. Harmon, and R. Fattal, “Efficient Simulation of Inextendible Cloth,” In*:* Proceedings of ACM SIGGRAPH, New York: ACM Press, pp. 281-290, 2007.

Received: May 17, 2014 Revised: July 21, 2014 Accepted: August 9, 2014