In the Proceedings of Computer Animation 2001, Seoul, Korea, November 2001.

Real-time Cloth Simulation with Sparse Particles and Curved Faces

Masaki Oshita Akifumi Makinouchi
Department of Intelligent Systems, Graduate School of Information Science and Electrical Engineering,
Kyushu University, Japan
moshita@db.is.kyushu-u.ac.jp

Abstract

In this paper, we present a novel technique for real-time cloth simulation. The method combines dynamic simulation and geometric techniques. Only a small number of particles (a few hundred at maximum) are controlled using dynamic simulation to simulate global cloth behaviors such as waving and bending. The cloth surface is then smoothed based on the elastic forces applied to each particle and the distance between each pair of adjacent particles. Using this geometric smoothing, local cloth behaviors such as twists and wrinkles are efficiently simulated. The proposed method is very simple, and is easy to implement and integrate with existing particle-based systems. We also describe a particle-based simulation system for efficient simulation with sparse particles. The proposed method has animated a skirt with rich details in real-time.

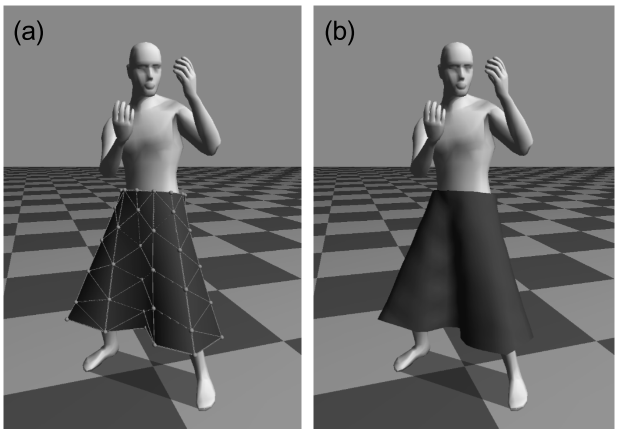

Figure 1. An example of the proposed approach. (a) A small number of particles (66 particles in this case) are controlled using dynamic simulation. (b) The cloth surface is then smoothed using geometrical techniques (1,944 faces were rendered).

1. Introduction

Physically based cloth simulation is an important current topic in computer animation. At present the most efficient and commonly used approach is particle-based cloth simulation. In particle systems, clothes are modeled as mass-distributed particles that are grouped into a fixed topology. In addition, forces that work on the particles, such as the fabric, gravity, and air forces are also modeled. Then, by solving a differential equation based on Newton's law, particle motions are computed. Currently this method is mainly used in off-line processes, such as for making motion pictures and real-time simulation is still difficult. One major problem is the large number of particles required for accurate simulation. Typically, from a few hundred to a hundred thousand particles are required to simulate a garment. Such a large number of particles needs a lot of computational time per step of the dynamic simulation and requires small time steps to avoid numerical instability. Although many techniques have been developed for fast and robust simulation, dynamic simulation requires too much computational time to animate clothes in real-time.

In this paper, a novel technique for real-time cloth simulation is presented. The method combines dynamic simulation and geometric techniques. Only a small number of particles (a few hundred at maximum) are controlled using dynamic simulation to simulate global cloth behaviors such as waving and bending. The cloth surface is then smoothed based on the elastic forces applied to each particle and the distance between each pair of adjacent particles. Using this geometric smoothing, local cloth behaviors such as twists and wrinkles are efficiently simulated (Figure 1).

The proposed method assumes that the motion of a cloth is computed using a coarse mesh. Therefore, the application of the proposed method to waving cloths, such as skirts or jackets, is appropriate because tight garments, such as sleeves or tight pants, need a fine mesh for dynamic simulation. In this paper, animation of a skirt is presented as an example. The proposed approach that has potential uses in computer games, real-time animation, virtual environments, and virtual fashion shows.

This balance of this paper is organized as follows; section 2 describes a geometrical smoothing method for generating a fine cloth mesh from a small number of particles. In section 3, a particle-based dynamic simulation system is described. Finally, an experimental result is presented and discussed.

2. Related Works

Smoothing of surfaces is a common technique in computer graphics. Many methods have been developed for modeling and compressing meshes, such as subdivision faces, parametric surfaces, and so on. These methods generate curved surfaces from a coarse mesh. However, just smoothing is not enough for generating a realistic cloth surface. For this, a kind of geometric method specifically for cloth surfaces is required.

Some researchers have developed heuristic models for computing a static cloth shape. These methods combine dynamics and geometric techniques. Taillefer [15] proposed a computational model for modeling the folds of a hanging cloth. Dhande et al. [4] modeled a wept surface. Kunii et al. [8] developed a formulation and deformation method for wrinkles. However, the deformation of wrinkles does not consider physical interactions with the underlying body. In these hybrid methods, an appropriate geometric cloth model is used to compute a specific and static cloth shape. Therefore, it is difficult to integrate them with an existing dynamic simulation system and animate them interacting with the changing environment. In this paper, we present a geometrical technique that is generically designed and easily integrated with standard particle-based cloth simulation systems.

Recently, Volino et al. [18] proposed a method for generating wrinkles using height maps. The method was expanded by Hadap et al. [7]. Using good looking wrinkle patterns provided by the designer as height maps, varying wrinkles are produced by modulating the height maps based on the compression of the cloth face similar to our method and the multi-layering of the height maps. Although this method generates realistic wrinkles, the pattern is basically designed for a specific posture. Therefore, it may produce a unnatural result under unexpected conditions, such as nonstandard posture of the underlying body, a strong wind, other external forces, constraints and so on. The method proposed in this paper produces a curved cloth surface based on the condition, is generically designed, and does not need extra data such as wrinkle patterns.

3. Geometric Smoothing for Cloth Surfaces

In this section, a geometric smoothing method is described. The smoothing method generates a fine cloth mesh based on a coarse cloth mesh that consists of a small number (typically a few hundred) of particles that are grouped into triangular faces. In smoothing a cloth surface, this method simulates the cloth's local behaviors within each original triangular area rather than just smoothing angular faces.

The proposed smoothing method consists of two stages; surface generation and deformation. Surface deformation involves particle normal control and edge length control. For surface generation, an existing method is used. Therefore, the original part of the smoothing method is the surface deformation method. In each simulation step, a cloth surface is rendered by the following steps;

1. The positions of all particles of the coarse cloth mesh are computed using dynamic simulation.
2. Surface generation. A cloth surface is generated by smoothing the triangular faces of the coarse cloth mesh.
3. Particle normal control. The generated surface is deformed so that its shape reflects the elastic forces applied to the particles.
4. Edge length control. The generated surface is deformed so that it keeps its original dimension by maintaining the length of each edge.
5. Triangular tessellation. The cloth surface is rendered as a number of small triangular faces.

3.1. Smoothing Triangular Faces

In the proposed method, PN triangles [16] are used to smooth triangular faces in a coarse cloth mesh. PN triangles substitute a three-sided cubic Bézier patch for each triangular face facilitating control of the smoothing mesh using the normal of the vertices (Figure 2). In addition, PN triangles ensure that the generated surface matches the original mesh on each vertex. These features are suitable for the control of the shape of clothes. Using PN triangles, the ten control points for representing a three-side cubic Bézier patch are computed from three positions, and from the normals of the vertices of the triangular face as illustrated by Figure 2.

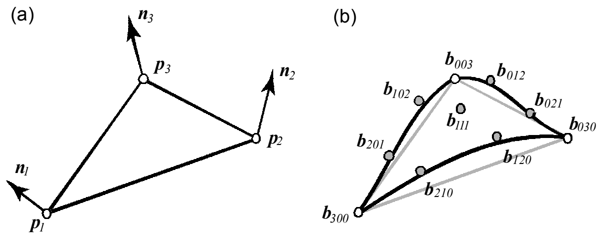

Figure 2. PN triangles replace a triangular face by a three-sided cubic Bezier patch computed from the three positions and normals. The cubic Bezier pathc is represented by coefficients or control points (B_{ijk}).

Many smoothing techniques have been proposed for modeling and compressing geometrical meshes. However, most smoothing techniques such as subdivision surfaces (e.g. Catmull-Clark [5]) and parametric surfaces (e.g. NURBS) generate a smoothed mesh from the topology of a coarse mesh and the positions of the vertices without considering their normals. Moreover, the generated surfaces are so smooth that they are away from the original vertices. These features make it difficult to control the shape of the produced surface. This is why we choose the PN triangles.

DeRose et al. [5] applied Catmull-Clark subdivision surfaces to cloth simulation. However, the subdivision surfaces were used for modeling clothes and the fine cloth meshes are produced for dynamic simulation. They discussed the problem regarding computing the continuous fabric parameters in a subdivision surface. Biermann et al. [3] proposed a subdivision surface technique with normal control. Although the shape of the generated surface can be controlled by the normal of its vertices, the generated surface is still too smooth and difficult to control. Volino et al. proposed SPHERIGON [17] that generates curved polygons based on the position and normal of the vertices of an original flat polygon in a similar way to PN triangles. SPHERIGON provides us the same facilities as do PN triangles that control the shape of a curved surface with their normal and ensure a curved face match on vertex positions. Therefore, SPHERIGON can also be used with the proposed surface deformation method. We have chosen PN triangles for first implementation since it just seems to be faster than SPHERIGON which involves a normalization process in computing a point on a curved polygon, and hardware accelerations for PN triangles [16] are expected. A more explicit comparison of the two methods in both visual results and computational time is a future topic.

3.2. Particle normal control

In particle-based systems, particles usually have no orientation. Only the position and velocity of particles are used in dynamic simulation. When the cloth faces are rendered using Gouraud shading, the particle normal is simply computed by taking an average of the normals of the adjacent faces.

Szeliski and Tonnesen [14] introduced oriented particles for a particle-based surface modeling system. They used oriented particles to keep the shape of surfaces in dynamic simulation. However, this approach makes the dynamic simulation more complex, and it is difficult to integrate with existing particle-based simulation systems. In our method, particles have no explicit orientation in dynamic simulation, and the particle normal is computed in the surface deformation step.

The proposed method computes particle normal so that the curved patch reflects the in-plane forces that are applied to each particle. Many force models such as elastic, bending, shearing, and twisting forces have been proposed[11]. Of these, elastic forces are the strongest and are responsible for the in-plane deformation of the clothes. Therefore, only elastic forces are considered in our implementation. Elastic forces are usually defined as a spring model in which a spring force works along each edge as follows.

$$f_{ij} = k_{elastic} \frac{\left|\boldsymbol{p}_i - \boldsymbol{p}_j\right|}{L_{ij}} \left(\left|\boldsymbol{p}_i - \boldsymbol{p}_j\right| - L_{ij}\right) \qquad (1)$$

where i, j is a pair of adjacent particles, $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ are the current positions of the two particles, L_{ij} is the distance between $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$ in their rest positions,, and k_{elastic} is the spring coefficient. The elastic force is computed for each edge and it works to make the distance between the two side particles close to the original distance.

Normal control is based on the principle of energy minimization, as illustrated in Figure 3. When the position of a particle is fixed, the normal on the particle is computed so as to minimize the energy on the particle (Figure 3(a)). When a particle receives an elastic force from an adjacent face, it can be assumed that the energy on the particle is proportional to the inner product between the elastic force and the normal vector on the particle (Figure 3(b)).

$$E_i^{\,j} = \boldsymbol{n}_i \cdot \boldsymbol{f}_i^{\,j} . \qquad (2)$$

The more parallel the normal vector is to the elastic force, the more the energy on the particle from the elastic force.

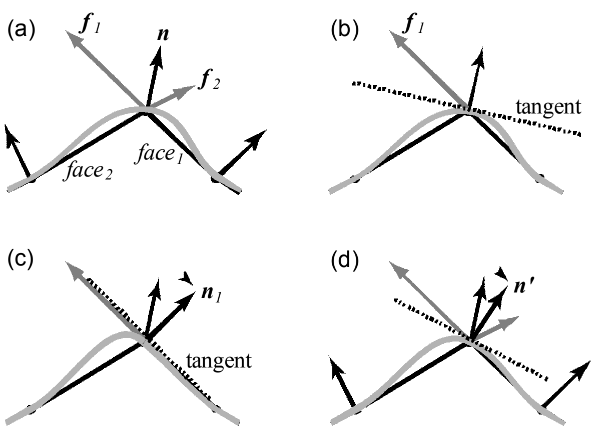

Figure 3. Normal control in section view. (a) The curved surface is computed based on the original normal using PN triangles. The elastic forces f_1 and f_2 are applied to $\boldsymbol{p}$ from $face_1$ and $face_2$ respectively. (b) The energy on $\boldsymbol{p}$ from f_1 is assumed to be proportional to inner product between f_1 and $\boldsymbol{n}$. (c) f_1 makes the tangent proportional to f_1, and $\boldsymbol{n}$ proportional $\boldsymbol{n}_1$ which is the normal of f_1. (d) The normal $\boldsymbol{n}'$ is computed by taking a weighted average.

Consequently, an elastic force on a particle works to make the tangent on the particle parallel to the elastic force (Figure 3(c)). Because a particle usually receives some elastic forces from all adjacent faces, to minimize the energy on the particle, the particle normal is computed by taking a weighted average of the normals of all adjacent faces with the magnitude of the elastic forces applied from each face (Figure 3(d)) as follows.

$$\boldsymbol{N}_i = \sum_{j=1}^{k} \boldsymbol{n}_j \frac{\left|\boldsymbol{f}_i^{\,j}\right|}{\sum_{j=1}^{k}\left|\boldsymbol{f}_i^{\,j}\right|}, \quad \boldsymbol{n}_i = \frac{\boldsymbol{N}_i}{\left|\boldsymbol{N}_i\right|} \tag{3}$$

where k is the number of adjacent faces, $\boldsymbol{n}_j$ is the normal on the jth face, $\boldsymbol{f}_i^j$ is the elastic force that applied to the ith particle from the jth adjacent face, and $\boldsymbol{f}_i^j$ is computed by taking the average of the two elastic force on the two edges on the jth face.

$$\boldsymbol{f}_i^{\,j} = \boldsymbol{f}_{ia} + \boldsymbol{f}_{ib} \tag{4}$$

where i, a, b are three particles in the jth triangle, and f_{ia} and f_{ib} are computed by equation (1).

3.3. Edge Length Control

In particle-based dynamic simulation, the length of an edge on a cloth mesh sometimes varies from its original length in a resting state because of the movements of particles and the constraints on the particles. When using a coarse cloth mesh, the variation of each edge length has a large influence on the entire cloth dimension because the initial edge length is longer than when using a fine cloth mesh. Therefore, each edge length is controlled to keep its original edge length.

When the length of a cubic Bézier curve is shorter than its original length (Figure 5(b)), the shape of the curve is deformed to make it longer while keeping the positions of the two terminal particles. To control a curve length, we first divide the curved patch into four curved patches (Figure 4). This divides each curve into two curves. The position of the midpoint $\boldsymbol{p}$ on the curve is then moved in the normal direction $\boldsymbol{n}$. The position $\boldsymbol{p}$ and the normal $\boldsymbol{n}$ on the midpoint is obtained from the original curved patch (Figure 5(c)).

$$\boldsymbol{p}' = \boldsymbol{p} + h\boldsymbol{n} \quad (h > 0) \tag{5}$$

In equation (5), h is computed so that the curve length is close to the original edge length. The divided four patches are also computed from the three corner particles and the three midpoints by again using PN triangles (Figure 5(d)).

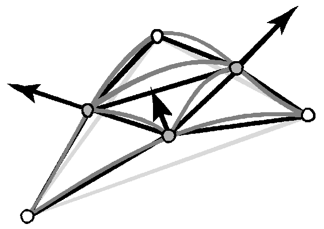

Figure 4. A cubic Bézier patch (Figure 2(b)) divided into 4 cubic Bézier patches. The 3 midpoints are obtained from the original PN triangle.

Unfortunately, the length of a cubic Bézier curve cannot be computed in a closed form. Therefore, the length should be computed numerically. By subdividing a Bézier curve into a number of segments and summing the length of all segments, we can approximate the curve length. Similarly, the edge length is controlled numerically and incrementally. Using a recursive bisection method [12], h in equation (5) is computed. We found the following equation to be a good approximation of the control distance as an initial guess.

$$h_0 = k\left(L_{ij} - \left|\boldsymbol{p}_i - \boldsymbol{p}_j\right|\right) \tag{6}$$

where L_{ij} is the initial edge length and k is an empirically determined parameter.

However, when a curve length is longer than its original edge length, it is difficult to make it shorter by using this

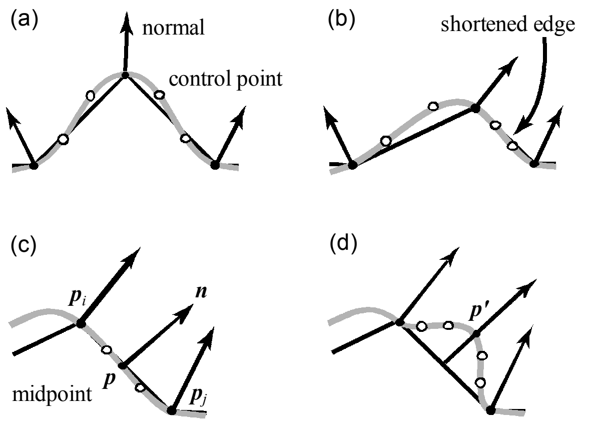

Figure 5. Edge length control in section view. (a) Original edges in the at rest length. (b) When an edge is shortened, (c) the midpoint of the shorter curve is controlled (d) so that the curve length is close to the original edge length.

approach. Therefore, in our system, instead of curve deformation in geometric smoothing, spatial constraints are applied in a dynamic simulation to avoid such elongated edges.

4. Dynamic Simulation with Sparse Particles

The smoothing method presented in the previous section is very simple, and is easy to integrate with an existing particle-based system because only particle positions are required for generating a smoothed surface. Many methods for particle-based simulation have been developed[11][20], but developing a successful particle-based system is still difficult. In developing a particle-based system, it is necessary to choose an appropriate method from many alternative methods for the purpose, trading-off between accuracy, speed and stability [2]. This section describes the chosen methods in our system and how they were modified to work efficiently with a small number of particles and the proposed smoothing method.

4.1. Numerical Integral method

Numerical integration is the essence of particle-based simulation and is the most important factor in the design of a particle-based simulation system. Using numerical integration, a partial differential equation is solved to compute the time-varying state of the particles.

There are two major methods for numerical integration: explicit and implicit. An explicit method was chosen for our implementation. Recently, implicit methods are widely used in many systems [1] [19], because they ensure stability which is the most important issue in interactive systems, and allow large steps instead of losing some accuracy. However, they have difficulties handling non-continuous constraints in dynamic simulation. On the other hand, explicit methods are a classic but accurate integral method. Each step of an explicit method is faster than an implicit method. Moreover, they are easily integrated with non-continuous constraints. However, a larger number of particles is used, so an explicit method requires smaller time steps. Therefore, as Baraff described [2], when many particles are used, explicit methods are slower than implicit methods and they become unstable.

However, when, only a few hundred particles are used, as here, an explicit method works faster than an implicit method. In addition, an explicit method is easily integrated with spatial constraints on penetrating particles and elongated edges that are introduced in our dynamic simulation. Therefore, an explicit numerical integral method, technically 4th-order Runge-Kutta, is currently used in our system. Further discussions about numerical integration are to be found in the literature [12][1].

4.2. Cloth Model

For modeling a cloth surface, triangular faces are used in our proposed method to apply the proposed smoothing method as explained in section 2. Elastic and bending force are introduced as internal cloth forces. The elastic force is described in section 3.2. The bending force is computed for each edge, and prevents bending between the two adjacent faces. In addition to the fabric forces, additional forces such as gravity, air velocity and damping forces [2] are also introduced. In some particle-based systems, square faces are used for representing a cloth instead of triangular faces. Here, diagonal forces such as shearing and twisting forces are used to prevent skews [11]. However, these are not introduced in our system because elastic and bending forces in triangular faces are considered to have similar affects on cloth surfaces.

4.3. Elastic Forces and Constraints

Clothes strongly resist stretching but allow bending, sharing or twisting. To realize such stiffness, the spring coefficient k_{elastic} in equation (1) should be of a large value. Otherwise, the cloth changes length and acts like a rubber or spring. Such a large coefficient value causes instability in numerical integration and requires very small time steps in dynamic simulation making the simulation slower. However, our system allows a comparatively small value of k_{elastic} because the edge length is maintained at its original length and large elastic forces are not required. Therefore, our system allows large time steps in dynamic simulation making the simulation faster.

In our system, two different methods are applied for handling shortened and elongated edges. As explained in section 3.3, when an edge length is shortened, the edge curve is deformed to keep the original edge length in geometric smoothing. However, this method cannot handle elongated edges. Therefore, when an edge length is elongated, spatial constraints are applied on the elongated edges in dynamic simulation in a way similar to that proposed by Provot [13]. If a curve length is longer than its original edge length, the positions of the two end-points of the elongated edge are directly altered to close the distance between the two particles.

4.4. Collision Detection and Constraints

Collision detection and response are also important issues in particle-based systems. To prevent a cloth penetrating the underling body contacts between the cloth and the body object should be detected and resolved. In our system, a simple state correction approach is used as explicit methods work well with this approach, while implicit methods require some extra techniques [2][19] to maintain stability.

If a fine cloth mesh with a large number of particles is used, the collisions between the cloth and any solid object can be handled by keeping the particles out of the solid object [2][6]. However, when using a coarse cloth mesh, as in our case, a cloth face may intersect with a solid object although the corner particles of the cloth face are kept away from the solid object. Therefore, in our system, we need to handle both vertex penetration and face intersection. Even though this needs extra computational time, using a coarse mesh makes collision detection and the related response faster.

For collision detection, a proximity query package (PQP) [9] a collision detection library is used in our system. PQP reports an intersection between a cloth face and a solid face. When an intersection is reported, an initial determination is made whether a vertex of the cloth face penetrates into the solid (Figure 6(a)) or the cloth face intersects with the solid but its vertices do not penetrate (Figure 6(b). When a cloth particle penetrates inside the solid, the position of the particle is corrected to the nearest point on the solid face (Figure 6(c)). When a cloth face intersects the object, the cloth face including its vertices is moved to the farthest point on the solid face (Figure 6(d)). When a particle contacts a solid face, the acceleration of the particle is directly constrained in dynamic simulation so that the relative velocity between the particle and the contacting face maintains the normal direction of the contacting face.

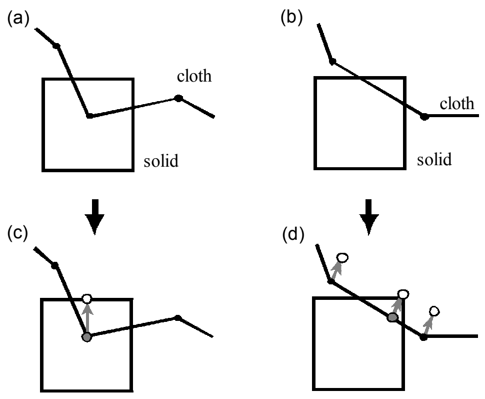

Figure 6. Collision detection and penetration avoidance. (a) Cloth vertex and solid penetration. (b) Cloth face and solid intersection. (c) Cloth particle or (d) cloth face position is corrected and constrained.

Collision detection and penetration avoidance is performed based on the original flat triangles. Therefore, the curved faces may penetrate into a solid object or the cloth itself. In terms of cloth-solid collisions, it is not a large matter because any generated surface basically matches the original mesh. To handle cloth-cloth collisions, collision detection between curved faces is required. Although a penalty-force approach could be a solution, this would involve large computational costs, and so a cloth-cloth collision detection method is yet to be introduced in our system.

5. Results

In this section an experimental result generated using the proposed method is presented and discussed. For the first test, we chose a skirt. Figure 7 shows the images from a generated animation of the skirt. The original skirt mesh used for the dynamic simulation consists of 66 particles grouped into 108 triangles. For the smoothed surface, each triangular face is rendered as 4 cubic Bézier patches using 18 small flat triangles. Consequently, 1,944 faces are tessellated using PN triangles and rendered. The skirt model is generated procedurally and

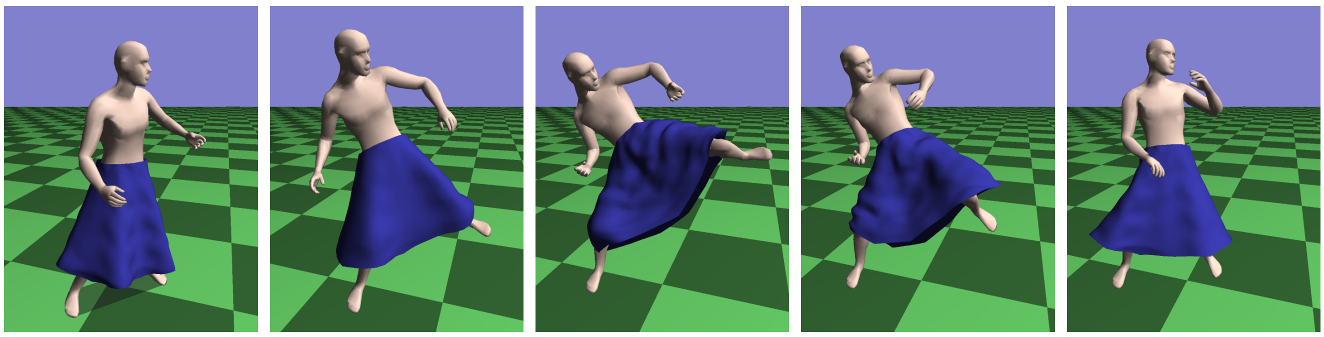
Figure 7. Images from an animation of a skirt.

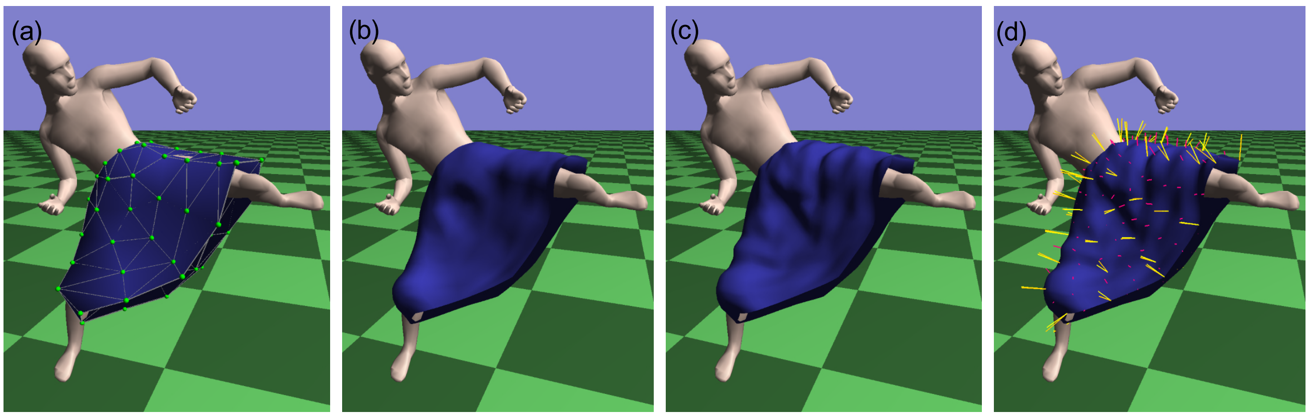

Figure 8. Smoothing surface in a frame of the animation. (a) Original mesh. (b) Smoothed surface using only PN triangles. (c) Smoothed surface using our proposed method. (d) Visualization of the surface deformation.

the figure animated in the motion-captured sequence of kicking.

Efficiency of the proposed method. Figure 8 shows smoothing stages in a frame of the animation. Figure 8(b) is the smoothed surface using only PN triangles and Figure 8(c) is the smoothed surface using our proposed method. Figure 8(d) shows the particle normal and edge length control used in Figure 8(c). Sticks in yellow and orange from the particles show the original and modified normals, respectively, and purple sticks show the edge length control. Although it is difficult to judge the visual results, it is found that the smoothed surface using our proposed method has rich details such as folds and twists around the left thigh in Figure 8(c), leading we believe to improved results. It is also found that the particle normal and edge length control are applied mainly on the left leg and on other shortened parts. When a cloth is relaxed and hanging naturally, the difference between the generated surface using only PN triangles and that using our proposed method is small. Our proposed method works more effectively when a part of the cloth is exposed to external constraints or is shortened.

Computational time and its optimizations. In the experiments, the duration between each simulation step was 1/30 seconds fixed. Each frame was divided into three steps in an explicit Eular integration. As a result, a 30 Hz animation was generated in real-time on a PC (PentiumIII 800MHz CPU + GeForce2MX video card). The computational time is shown in Table 1. For the proposed smoothing method, 2.4 extra milli-seconds were used compared to when using only PN triangles. This means that, by allowing a few extra

Used method	Computational time (ms / frame)				Speed (fps)
	Integration	Collision	Smoothing	Total	
Proposed method	3.5	14.7	6.4	23.7	29.8
PN triangles	3.6	14.6	4.0	21.9	32.1

Teble 1. Maximum computational time per frame in the animation of skirt when our proposed method or only PN triangles were used. The computational time for cloth simulation (numerical integration, collision handling, and smoothing surface) was about 60% of the entire computational time.

mill-seconds per a hundred faces in addition to that allowed for the PN triangles, more realistically deformed cloth surfaces are achieved. Although the triangular tessellation is currently computed in the CPU and the tessellated triangles are stored in memory, hardware supports for PN triangles [16] could accelerate the smoothing method. Since the program used in our experiments has not been optimized yet, the computational time in table 1 could become faster after optimization for both numerical computation and rendering.

6. Conclusion and Future Work

The smoothing method presented in this paper is very simple and efficiently adds rich details on a coarse cloth mesh. However, the generated surfaces are slightly limited. The edge length control adds at most one fold on the center of a shortened edge and the direction of the fold is determined from just the normal of the two end points. To generate more realistic surfaces, the number, position, and direction of the folds should be controlled. This we think might be possible by taking into account bending forces and compression speed. For more subtle and more wrinkles, wrinkle patterns provided by a designer [7] or pre-computed using dynamic simulation [10], could be appropriate and could be integrated with our proposed method. However, such wrinkle patterns require extra modeling processing or pre-computation time, and cannot produce appropriate wrinkles under unexpected conditions. The goal of our research has been to provide a geometric method that is easily usable and generates cloth surfaces under any conditions using heuristic methods that capture the characteristics of clothes rather than using static and expensive wrinkle patterns.

We intend to apply our proposed method to garments such as jackets, shirts and pants in which the interaction with the underlying body is more complex and critical. To simulate a difficult interaction, such as the figure putting on a garment or picking up a cloth, while keeping a fast and robust simulation, an adaptive technique that inserts and removes appropriate particles into the cloth mesh would be required.

Currently only coarse cloth meshes achieve real-time performance. Many on-line applications such as computer games require real-time cloth simulation at a small computational cost. We believe that our proposed methods that add detail on a physically simulated coarse cloth mesh are an effective and reasonable approach.

Acknowledgements

The work was partially supported by the Grant-in-Aid for Scientific Research (10308012) from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

[1] David Baraff and Andrew Witkin , "Physically Based Modeling: Principles and Practice", SIGGRAPH '97 Course Notes,1997. (http://www.cs.cmu.edu/~baraff/sigcourse/index.html)

[2] David Baraff, and Andrew Witkin, "Large Steps in Cloth Simulation", Proc. of SIGGRAPH '98, pp. 43-54, 1998.

[3] Henning Biermann, Adi Levin, and Denis Zorin, "Piecewise Smooth Subdivision Surfaces with Normal Control", Proc. of SIGGRAPH 2000, pp. 113-119, 2000.

[4] S. G. Dhande et al., "Geometric Modeling of Draped Fabric Surface", Proc. of IFIP International Conference on Computer Graphics, pp. 349-356, 1993.

[5] Tony DeRose, Michael Kass, and Tien Truong, "Subdivision Surfaces in Character Animation", Proc. of SIGGRAPH '98, pp. 85-94, 1998.

[6] Bernhard Eberhardt, Andreas Weber, and Wolfgang Strasser, "A Fast, Flexible, Particle-System Model for Cloth Draping", IEEE Computer Graphics and Applications, vol. 16, no. 5, pp.52-59, September 1996.

[7] Sunil Hadap, Endre Bangerte, Pascal Volino, and Natia Magnenat-Thalmann, "Animating Wrinkles on Clothes", Proc. of IEEE Visualization '99, pp. 175-182, 1999.

[8] T. L. Kunii, and H. Gotoda, "Modeling and Animation of Garment Wrinkle Formation Processes", Proc. of Computer Animation '90, pp. 131-147, 1990.

[9] Eric Larsen, Stefan Gottschalk, Ming C. Lin, and Dinesh Manocha, "Fast Proximity Queries with Swept Sphere Volumes", Technical report TR99-018, Department of Computer Science, University of N. Carolina, Chapel Hill, 1999. (http://www.cs.unc.edu/~geom/SSV/)

[10] Daniel L. Herman, "Using Precomputed Cloth Simulations for Interactive Applications", SIGGRAPH 2001 Sketches and Applications, p. 273, 2001.

[11] Hing N. Ng and Richard L. Grimsdale, "Computer Graphics Techniques for Modeling Cloth", IEEE Computer Graphics and Applications, vol. 16, no. 5, pp.28-41, September 1996.

[12] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery, "Numerical Recipes", Cambridge University Press, 1992. (http://www.nr.com/nronline_switcher.html)

[13] Xavier Provot, "Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior", In Proc. Graphics Interface '95, pp 147-154, 1995.

[14] R. Szeliski, and D. Tonnesen, "Surface Modeling with Oriented Particles", Computer Graphics (Proc. of SIGGRAPH '92), Vol. 26, No. 2, pp 185-194, 1992.

[15] F. Taillefer, "Mixed Modeling", Proc. Compugraphics, pp. 467-478, 1991.

[16] Alex Vlachos, Jorg Peters, Chas Boyd, and Jason L. Mitchell, "Curved PN Triangles", Proc. of the 2001 ACM Symposium on Interactive 3D Graphics, 2001. (http://alex.vlachos.com/graphics/)

[17] Pascal Volino, and Nadia Magnenat-Thalmann, "The SPHERIGON: A Simple Polygon Patch for Smoothing Quickly your Polygonal Meshes", Proc. of Computer Animation '98, pp. 72-78, 1998.

[18] Pascal Volino, and Nadia Magnenat-Thalmann, "Fast Geometrical Wrinkles on Animated Surfaces", Proc. of the 7-th International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media '99 (WSCG '99), 1999.

[19] Pascal Volino, and Nadia Magnenat-Thalmann, "Implementing Fast Cloth Simulation with Collision Response", Proc. of Computer Graphics International 2000, 2000.

[20] Pascal Volino, and Nadia Magnenat-Thalmann, Virtual Clothing – Theory and Practice, Springer-Verlag, 2000.