DYNAMIC SIMULATION OF INEXTENSIBLE CLOTH

Jan Bender, Daniel Bayer and Raphael Diziol

Institut für Betriebs- und Dialogssysteme
Universität Karlsruhe
Am Fasanengarten 5
76128 Karlsruhe
Germany

ABSTRACT

In this paper an impulse-based method for cloth simulation is presented. The simulation of cloth is required in different application areas like computer animation, virtual reality or computer games. Simulation methods often assume that cloth is an elastic material. With this assumption the simulation can be performed very efficiently using spring forces. The problem is that many textiles cannot be stretched significantly. A realistic simulation of these textiles with spring forces leads to stiff differential equations which cause a deterioration of performance. The impulse-based method described in this paper solves this problem and allows the realistic simulation of inelastic textiles.

KEYWORDS

Cloth simulation, physically-based modelling, impulse-based simulation, inelastic textiles

1. INTRODUCTION

The dynamic simulation of cloth is an important topic in computer graphics. It has different application areas, like computer animation, games or fashion design.

Many simulation methods assume that textiles are elastic. This assumption is made due to performance reasons. If a simulated textile is elastic, a mass-spring system can be used in order to perform an efficient simulation. For example, Georgii and Westermann (2005) describe a method for a fast dynamic simulation of mass-spring systems on the GPU. Other works use an energy function that describes the deformation of the model in order to compute forces between the mass points in the model. The simulation with forces is very easy and fast, since no constraints must be resolved. The problem of these methods is that the strain of the model cannot be limited.

Many textiles do not stretch significantly under their own weight and are not very elastic. A realistic simulation of these textiles with methods based on the computation of deformation forces is hard. For the simulation of inextensible textiles very stiff springs or huge forces are required. These forces lead to stiff differential equations which decrease the stability and the performance of the simulation.

This paper presents a constraint-based method for the simulation of inextensible cloth. In the simulation a piece of cloth is represented by a mesh of particles that are linked by distance constraints. It is shown how these constraints can be resolved by an impulse-based method. This method can guarantee a maximum strain of the cloth in contrast to the methods based on the computation of forces. Therefore, a realistic simulation of inextensible textiles is possible.

In the next section related work is discussed. Afterwards the impulse-based approach is presented. First the simulation of particles and distance constraints is described. Then a dynamic model for the simulation of inextensible cloth is introduced. This model consists of particles that are linked by distance constraints and spring dampers. The constraints make the model inextensible and the spring dampers are used to simulate shearing and bending. In the end of the section it is shown how the dependencies between the constraints in the mesh are resolved. Results of the presented method are presented in section 4. The last section concludes the work.

2. RELATED WORK

The simulation of cloth and fabrics is an import area of research in computer graphics. This area has a long history of research since the first work about the simulation of deformable models was presented by Terzopoulos et al. (1987). House and Breen (2000) and Magnenat-Thalmann and Volino (2005) give a good survey of the research done in cloth animations. Current research problems in clothing simulation are summarized by Choi and Ko (2005).

Cloth is often simulated as an elastic material due to performance reasons but many textiles are not noticeably stretchable. Many approaches use spring forces for the simulation of stretchable cloth. In fact today's clothing simulations are mainly based on spring-mass systems of particles (see, e.g. Choi and Ko (2002)). The realistic simulation of an inextensible piece of cloth with such an approach would require large spring constants. This leads to stiff differential equations which are hard to integrate and decrease the numerical stability (Hauth et al. (2003)). Stiff differential equations can only be solved by reducing the time step size or by special integration methods. In both cases the performance of the simulation is decreased significantly. Hence, the methods based on the computation of forces are not suitable for simulating inextensible textiles.

Xavier Provot (1995) restricts the strain of the cloth model to a certain limit by using a post-processing algorithm. This algorithm moves linked particles to new positions if their current distance exceeds 10 percent of their original distance. The problem of such displacements is that self-intersections can occur. Robert Bridson et al. (2002) avoid this problem by using a different post-processing method which also resolves contacts and collisions with friction.

Motivated by the problems with explicit integration methods, David Baraff and Andrew Witkin (1998) used implicit integration for a stable simulation. Later on, other works studied the use of implicit integration as well (see, e.g. Pascal Volino and Nadia Magnenat-Thalmann (2001)). Implicit integration methods increase the stability of the simulation significantly. Therefore, large simulation steps can be performed. But the use of implicit integration has also a disadvantage. In general, a non-linear system of ODEs has to be solved in each simulation step.

Another approach is to use constraints to enforce the conditions instead of integrating the spring forces directly (see, e.g. Fuhrmann et al. (2003)). Especially, if small tolerances are demanded this approach provides a good alternative. A system for constrained-based simulation of inextensible cloth based on the Lagrangian mechanics is given in Goldenthal et al. (2007).

In this paper a new constraint-based approach is presented that uses impulses for the simulation of cloth. It is shown that the use of impulses instead of spring forces allows an accurate solution of the constraints.

3. CLOTH SIMULATION

A piece of cloth is represented by a mesh of particles that are linked by distance constraints. The following sections describe the dynamic simulation of particles and constraints and the handling of their dependencies in a mesh.

3.1 Particle simulation

A particle is a body that has a mass but no volume. Since a particle has no dimension, it has just translational degrees of freedom and no rotational ones. In the simulation the state of a particle is described by its mass *m* its position *c* and its linear velocity *v*. In general the mass of a particle is constant during the simulation. A time step of a particle is performed by integrating its velocity and its position over time. It is assumed that the sum of all external forces acting on a particle, like e.g. gravity, is constant during the time step. In this case the velocity and position after a step of size *h* can be computed directly by the following equations:

$$v(t_0 + h) = v(t_0) + \int_0^h \frac{F_{\text{ext}}}{m}\, dt = v(t_0) + \frac{F_{\text{ext}}}{m} h \,.$$

$$c(t_0 + h) = c(t_0) + \int_0^h v(t_0) + \frac{F_{\text{ext}}}{m} t \; dt = c(t_0) + v(t_0)\, h + \frac{1}{2}\frac{F_{\text{ext}}}{m} h^2 \,.$$

where F_{ext} is the sum of all external forces and t_0 is the start time of the simulation step. In the simulation it is differentiated between dynamic and static particles. In contrast to dynamic particles static ones have a fix position and no velocity.

3.2 Distance constraints

A distance constraint for two particles *a* and *b* consists of two parts: a position constraint and a constraint for the velocities of the particles. The position constraint is defined as follows:

$$C(a,b,t) = d(t) - d(0) = 0 \quad \text{with} \quad d(t) = \left| c_a(t) - c_b(t) \right| .$$

This means that the difference of their actual distance and their distance at the beginning of the simulation must be zero to satisfy this constraint. Hence, their distance must stay constant over time. In the simulation this constraint is satisfied by impulses. At time *t* these impulses are computed by using a preview of the simulation step. For a distance constraint the distance of the corresponding particles *d(t+h)* after the next simulation step of size *h* is determined by integrating the particle positions (see section 3.1). The difference between the value *d(t+h)* and the distance *d(0)* at the beginning of the simulation is exactly the error that would occur if the simulation step would be performed without regarding the position constraint:

$$e = d(t + h) - d(0).$$

This error can be eliminated by computing impulses and applying them at the beginning of the simulation step at time *t*. Since an unconstrained particle has a linear motion, the two particles must change their relative velocity by e/h in direction of the constraint in order to change their distance by *e* in a time step of size *h*. The direction of the constraint is given by the vector from particle *a* to particle *b*. In order to satisfy the conservation of momentum two impulses *p* and *–p* of the same magnitude and opposite directions are applied to perform this velocity change. The impulse *p* can be determined by solving the equation:

$$\Delta v_a(p,t) - \Delta v_b(-p,t) = (c_b(t) - c_a(t))\frac{e}{h}$$

where $\Delta v_a(p,t)$ is the velocity change of particle *a*, when the impulse *p* is applied. The differentiation between dynamic and static particles is done using the following value for computing the velocity change:

$$k_a = \begin{cases} 1/m_a & \text{if particle } a \text{ is dynamic} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

The resulting equation for the impulse is:

$$\Delta v_a(p,t) - \Delta v_b(-p,t) = k_a p - k_b(-p) = (k_a + k_b)p = (c_b(t) - c_a(t))\frac{e}{h} \tag{2}$$

which can be solved if at least one of the particles is dynamic and the particles have different positions. It is assumed that the constraint is satisfied at the beginning of a simulation step, so the particles cannot have the same positions. After the determination of the impulse p it must be applied to the particles in positive and negative direction respectively. Due to the resulting velocity change, the position constraint will be satisfied after the simulation step.

The second part of the distance constraint is a constraint for the velocities of the particles:

$$C(v,t) = (v_b(t) - v_a(t))(c_b(t) - c_a(t)) = 0 .$$

This means that the relative velocity of the particles in direction of the constraint must be zero. In contrast to the position constraint, here the required impulse can be determined directly without a preview because an impulse causes an instantaneous velocity change. The impulse that satisfies the constraint is computed by:

$$p = \frac{(v_b(t) - v_a(t))(c_b(t) - c_a(t))}{k_a + k_b} . \tag{3}$$

The computation of an impulse in order to correct the velocities is not absolutely necessary for the simulation of a distance constraint. A higher degree of accuracy can be achieved by regarding the velocity constraint in the simulation but at the cost of performance.

3.3 Cloth model

The model that is used for the dynamic simulation of inextensible cloth consists of particles that are linked by distance constraints. These constraints define a mesh which represents the simulated cloth. Figure 1 shows the mesh for a quadratic piece of cloth.

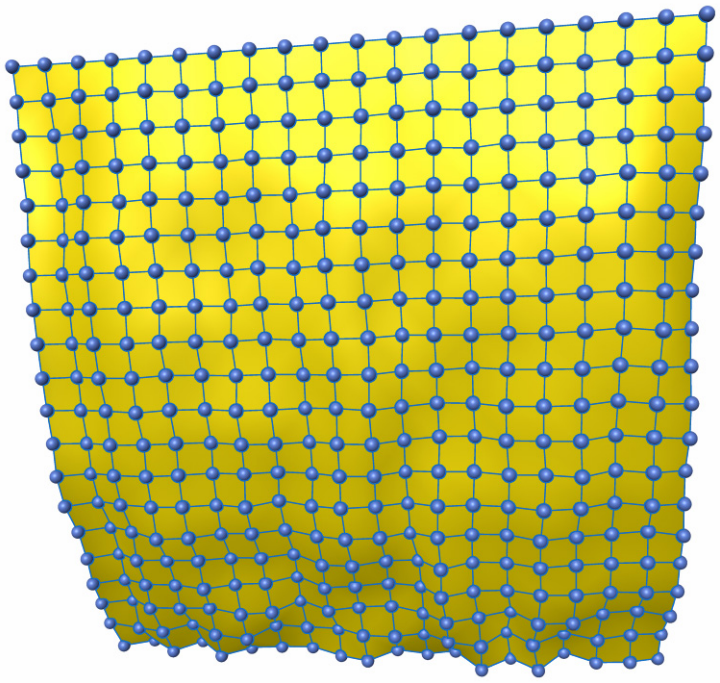

Figure 1. Particle model with distance constraints used for the simulation of cloth

The distance constraints are used to limit the maximum strain of the simulated textile. In order to simulate shearing and bending forces springs are added to the model. Figure 2 shows these springs for a particle with the position (i, j) in the model. This particle is linked to all its diagonal neighbours by shearing springs and to the particles with the positions $(i+2, j)$, $(i-2, j)$, $(i, j+2)$ and $(i, j-2)$ by bending springs.

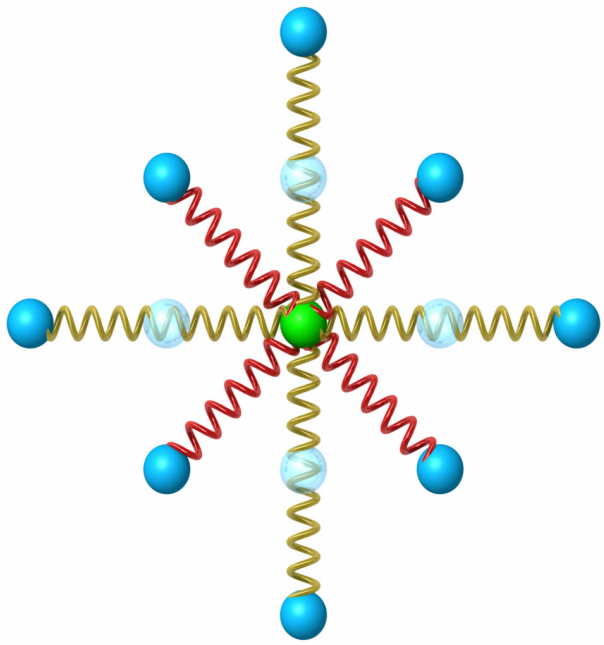

Figure 2. The particle in the middle is linked to its diagonal neighbours by shearing springs (red). The horizontal and vertical springs (yellow) are used to simulate bending forces.

As mentioned in section 3.1, the external forces acting on a particle should be constant for a simulation step. Therefore, the spring forces are integrated at the beginning of a simulation step using an explicit Euler integration. Then, the velocities of the particles are changed directly by the resulting impulses.

3.4 Mesh simulation

In the simulation cloth is represented by a mesh. In each vertex of this mesh one particle is created. A simple approach to simulate stretchable cloth is to introduce a damped spring on each edge of the mesh. The elasticity of the cloth is controlled by the spring constant and the damping coefficient. However, the simulation of inextensible cloth using spring forces leads to stiff differential equations which cause a significant decrease of performance.

In order to simulate cloth using impulses a distance constraint as described above is defined for each pair of particles that are connected by an edge in the mesh. The constraint is satisfied if the distance between the positions of the two corresponding particles equals the length of the edge at the beginning of the simulation. If each constraint is exactly satisfied, the simulated mesh is inextensible.

In such a mesh a particle is linked by multiple distance constraints to other particles. The impulses of constraints which have a common particle influence each other. The resulting dependencies can be resolved in different ways. In the following an iterative method and a method based on a system of linear equations are presented in order to resolve the dependencies.

3.4.1 Iterative method

Dependencies in the constraint structure can be resolved if the constraints are handled in an iterative process. In an iteration step an impulse is computed for each distance constraint as described in section 3.2. Since the impulses influence each other the constraints are not satisfied within one step but the errors are reduced. The iterative process stops, when all constraints are satisfied within a predefined tolerance. Schmitt et al. (2005) proofed that this iterative method converges to the physically correct solution.

Algorithm 1 shows how a simulation step with the iterative method is performed. First the impulses for the position constraints are determined. Then the new positions of the particles are computed by integration. In the last step impulses are determined in order to satisfy the velocity constraints of the model.

```
Input: List of all position constraints  C^pos = {C_0^pos, ..., C_l^pos}
       List of all velocity constraints  C^vel = {C_0^vel, ..., C_m^vel}
       List of all particles  P = {p_0, ..., p_n}
       Tolerance value  ε

// Compute the correction impulses for all position constraints
repeat
    allConstraintsSatisfied = true
    for i = 0 to l
        determine  C_i^pos(t+h) = e_i = d_i(t+h) − d_i(0)  by integration (preview)
        if  e_i ≥ ε
            compute correction impulse by solving equation 2
            compute new velocities of linked particles for time t
            allConstraintsSatisfied = false
until allConstraintsSatisfied

// Time step from t to t+h
for i = 0 to n
    integration of the state of particle  p_i

// Compute the correction impulses for all velocity constraints
repeat
    allConstraintsSatisfied = true
    for i = 0 to m
        if  C_i^vel(t+h)  is not satisfied
            compute correction impulse by equation 3
            compute new velocities of linked particles for time t+h
            allConstraintsSatisfied = false
until allConstraintsSatisfied
```

Algorithm 1: Simulation step with the iterative method

The iterative method has many advantages. It is very simple to implement. Collision and contact handling with friction can be easily integrated in the iterative process (Bender et al., 2006). The maximum allowed extensibility of the simulated cloth is directly defined by the used tolerance value. The iterative process can be interrupted at any time to get a preliminary result. Even if the process is interrupted and the tolerance value is not reached, the simulation stays stable. All constraints which have no common particle and therefore, no direct dependency can be solved in parallel (Bayer et al. 2009).

3.4.2 SLE method

The disadvantage of the iterative approach is that the simulation of a complex mesh requires many iteration steps if a small tolerance value is used. Because of this another method was developed for simulating totally inextensible cloth.

The dependencies in the cloth model can be described by a system of linear equations (SLE). The impulses of all constraints are determined simultaneously by solving the SLE. The SLE describes the structure of the constraints in the model. Therefore, the constraints must not be redundant. Otherwise the SLE is overdetermined and the simulation can get instable. Figure 3 shows the degrees of freedom for a small cloth model. The particles are examined from the upper left to the lower right. The first particle can choose its position freely and has three degrees of freedom. Its direct neighbours can move on the surface of a sphere around it. Therefore, they have two degrees of freedom. The same is valid for the particles on the left and upper border. The rest of the particles have one degree of freedom, since they can only move on a circle. Hence, the model has 15 degrees of freedom. Each free particle has three degrees of freedom and a system of nine particles has 27 degrees of freedom. Therefore, a SLE for the model in the figure which is not overdetermined must have the dimension twelve in order to remove twelve degrees of freedom. Since there are twelve distance constraints in the model this condition is satisfied. But this is not generally true. If a quad of four particles degenerates to a line, the SLE gets overdetermined. Such a situation does not occur in praxis

due to the fact that there act impulses on the diagonal of each quad in order to simulate shearing forces. However, if such a situation occurs, it can be resolved by slightly moving the diagonal particles apart from each other.

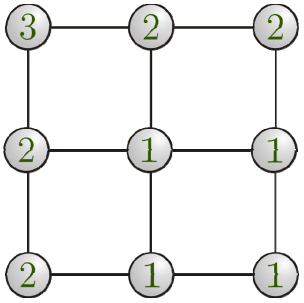

Figure 3.Degrees of freedom in the cloth model

The system of linear equations to compute the impulses of all constraints has the following form:

$$A\ p = \Delta v\ .$$

The matrix A contains a row and column for each distance constraint. It describes how the impulses of the vector p change the velocities of the particles. The vector Δv contains all velocity changes that are required to resolve the constraints. The complete constraint structure of the model is represented by the matrix A. Therefore, the dependencies between the constraints are taken into account, when solving the SLE in order to determine the impulses.

The same SLE can be used for position and velocity constraints. Just the vector Δv is different in both cases. Hence, the i-th element of the vector is defined as follows:

$$\Delta v_i = \begin{cases} \frac{e_{\text{pos},i}}{h} & \text{if } i \text{ is a position constraint} \\ e_{\text{vel},i} & \text{if } i \text{ is a velocity constraint} \end{cases}$$

where $e_{\text{pos},i}$ and $e_{\text{vel},i}$ are the position and velocity error respectively.

Each row and column in the matrix A represents a distance constraint. An element $A_{i,j}$ of the matrix describes how the constraint i depends on constraint j. This element is not zero if and only if both constraints have a common particle. Otherwise there is no direct dependency between them. For the computation it is important if the common particle is the first or second particle of the particular constraint. This influences the sign of the corresponding matrix element. The following case differentiation describes this relation:

$$B_{i,j} = \begin{cases} k_{i_1} & \text{if } i_1 = j_1 \wedge i_2 \neq j_2 \\ k_{i_2} & \text{if } i_2 = j_2 \wedge i_1 \neq j_1 \\ -k_{i_1} & \text{if } i_1 = j_2 \wedge i_2 \neq j_1 \\ -k_{i_2} & \text{if } i_2 = j_1 \wedge i_1 \neq j_2 \\ k_{i_1} + k_{i_2} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

where i_1 and i_2 are the indices of the first and second particle of the i-th constraint. The values k_i are determined using equation 1. The diagonal elements are special, since i and j are the indices of the same constraint. Therefore, i and j have two common particles and their k values are added.

If two constraints have a direct dependency, the impulses of both constraints influence their common particle. Because of this all dependencies must be regarded, when computing a correction impulse. For the computation of an impulse for constraint i the impulses of the dependent constraints must be projected into the space of constraint i. The required projection matrix of a distance constraint is defined as follows:

$$P_i = (c_{i_2}(t) - c_{i_1}(t)) .$$

The matrix A of the SLE is determined by projecting the matrices $B_{i,j}$ using the projection matrices of the corresponding constraints:

$$\begin{pmatrix} B_{1,1} & P_1 B_{1,2} P_2^T & \cdots & P_1 B_{1,n} P_n^T \\ P_2 B_{2,1} P_1^T & B_{2,2} & \ddots & \vdots \\ \vdots & \ddots & \ddots & P_{n-1} B_{n-1,n} P_n^T \\ P_n B_{n,1} P_1^T & \cdots & P_n B_{n,n-1} P_{n-1}^T & B_{n,n} \end{pmatrix} \begin{pmatrix} \tilde{p}_1 \\ \tilde{p}_2 \\ \vdots \\ \tilde{p}_n \end{pmatrix} = \begin{pmatrix} \Delta v_1 \\ \Delta v_2 \\ \vdots \\ \Delta v_n \end{pmatrix}$$

where $\tilde{p}_i$ is the magnitude of the i-th correction impulse. The diagonal elements do not need a projection because $P_i P_i^T = 1$. The three-dimensional impulse is computed by multiplying the magnitude with the transposed projection matrix:

$$p_i = \tilde{p}_i P_i^T .$$

In the end each computed impulse is applied to the particles of the corresponding constraint in opposite directions in order to guarantee the conservation of momentum.

The required correction impulses are determined at once by solving the SLE. Since all dependencies are regarded, the impulses solve the constraints exactly. Hence, totally inextensible cloth models can be simulated with this method. The matrix A is constant at time t. Because of this, the factorization of the matrix has to be done only once per simulation step. The same factorization can be used for resolving the velocity constraints and the position constraints of the next simulation step, since their impulses are computed at the same point of time. This is an important fact because the factorization is the most time-consuming part of a simulation step.

4. RESULTS

This section presents results of the impulse-based methods. The introduced simulation methods were implemented in C++. Each simulation in this section is performed on a PC with a 2.4 GHz Intel Core 2 Quad processor.

The tolerance values of the iterative approach define the maximum allowed strain of the simulated textile. In figure 4 a model with a grid of 40×40 particles is shown. The particles are linked by 3120 distance constraints. The two particles in the top corners are static in order to study the relaxation of the cloth due to gravity. Tolerance values between 0.1 and 0.0001 are used to simulate a maximum allowed strain between 10% and 0.01%. The presented iterative method supports even smaller values. In contrast to the iterative approach the SLE method can only simulate totally inextensible textiles.

A regular cloth model is used for the performance tests (see section 3.3). Models with different sizes are simulated in order to measure the performance of the introduced methods. The smallest model has a grid of 10×10 particles and the largest model has 50×50 particles. The models have between 180 and 4900 distance constraints which must be resolved by impulses. For shearing and bending between 241 and 7201 spring dampers are required. The performance of the iterative approach does not only depend on the complexity of the model, it also depends on the maximum allowed strain. Therefore, different tolerance values are used in order to measure the performance of this method.

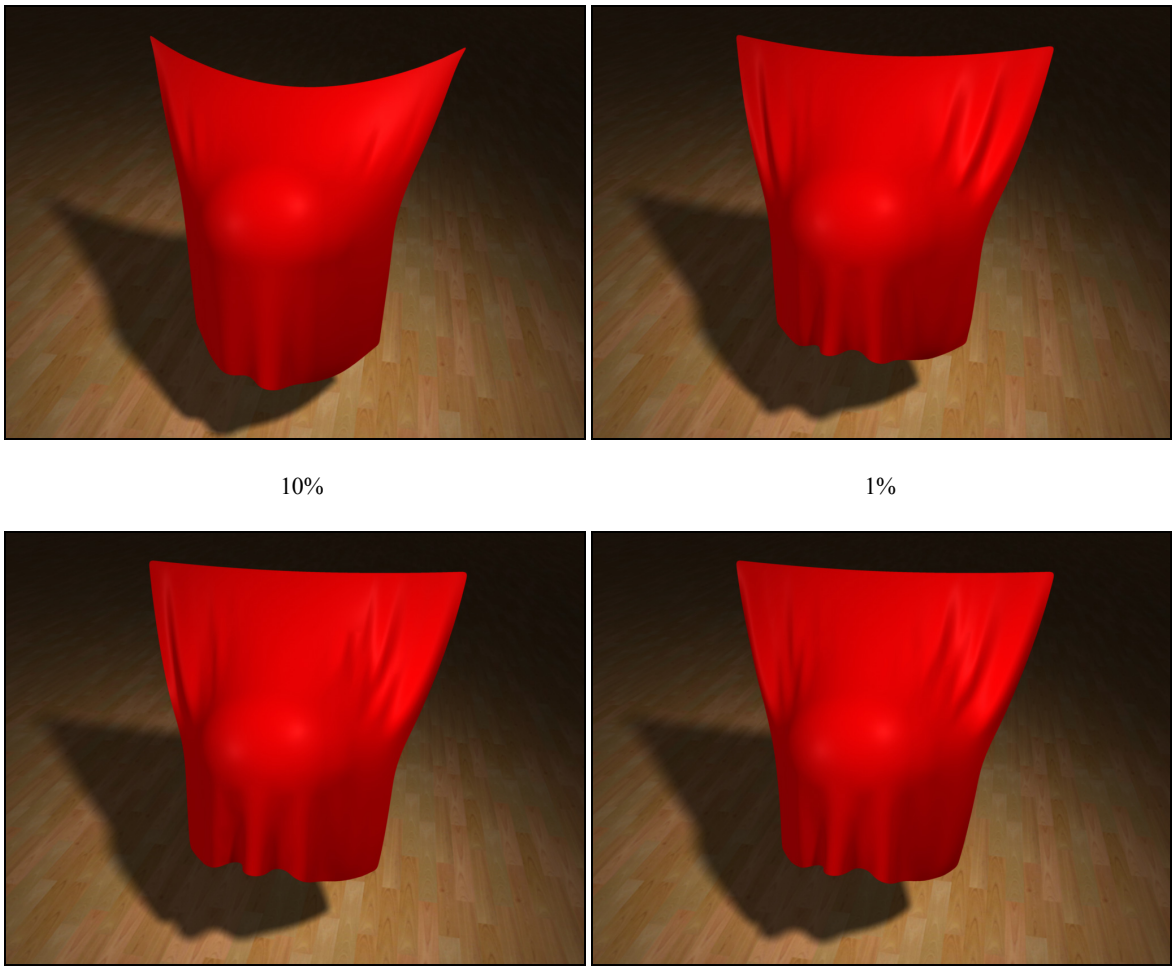

Figure 4. Simulation of a cloth model with different tolerance values in order to get a maximum allowed strain between 0.01% and 10%

A time step size of h=1/30 s is used for the simulation to get a frame rate of 30 frames per second. The cloth model must move in order to measure the performance under realistic conditions. Therefore, the model is parallel to the floor at the beginning of the simulation. Hence, it will swing around the static particles. To prevent a uniform motion the velocities of some randomly chosen particles are set to different values before the simulation starts. The average computation time per frame is determined after 500 simulation steps.

Figure 5 shows computation times for the models simulated using the iterative method with different tolerance values. The horizontal line marks the time step size of 1/30 s. Everything below this line is simulated faster than real-time.

Even for a maximum allowed strain of 0.01% the simulation of the 30×30 mesh is faster than real-time. The smallest model is about three times faster than real-time using the smallest tolerance value of 0.0001. The model consisting of 50×50 particles linked by 4900 distance constraints can be simulated in almost real-time, when the simulation is performed with a maximum allowed strain of 1%. If the smallest tolerance value is used, the largest model is about six times slower than real-time.

The performance of the simulation can be increased if the iterative process is stopped after a given maximum number of iterations. This has the advantage that the maximum computation time for each step is known. The problem is that the tolerance values are not reached when interrupting the iterative process. In this case the maximum allowed strain cannot be guaranteed but the simulation is still stable and the results

are visually plausible. By reducing the maximum number of iterations to five, the 50×50 mesh can be simulated about 2.3 times faster than real-time.

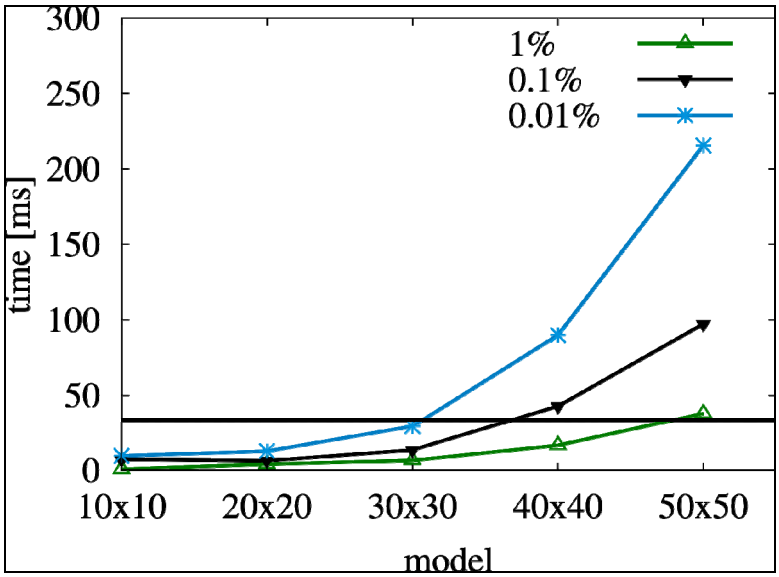

Figure 5. Average computation times per simulation step for cloth models of different sizes simulated with the iterative approach and a maximum allowed strain between 10% and 0.01%

The same five models were simulated again using the SLE method. Figure 6 presents the average computation times per simulation step for this method. The simulation gets slow for complex models since the system of linear equations that must be solved in each simulation step gets very large. For the largest model the system of linear equations has a dimension of 4900.

The smallest model can be simulated about four times faster than real-time but already the model with a 20×20 mesh is four times slower than real-time. The SLE method requires much computation time but it can simulate totally inextensible textiles. The constraints are not resolved within a certain tolerance. They are satisfied exactly in each step.

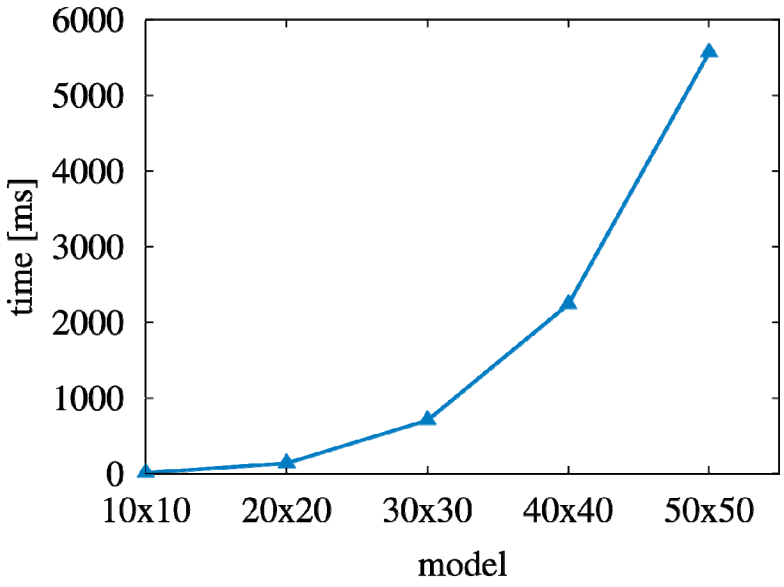

Figure 6. Average computation times per simulation step of the SLE method

For the accurate handling of collisions and contacts with friction the constraint-based collision response method of Bender et al. (2006) was integrated in the iterative process described above. Figure 7 and 8 show the results of two different simulations with the impulse-based method described in this paper. The tolerance value used in the iterative process defines the maximum extensibility of the simulated cloth. In both simulations the chosen value prevented the cloth from stretching more than 0.01 percent. The introduced method also works with much smaller tolerance values but at the cost of performance. Both simulations were performed with collision and contact handling.

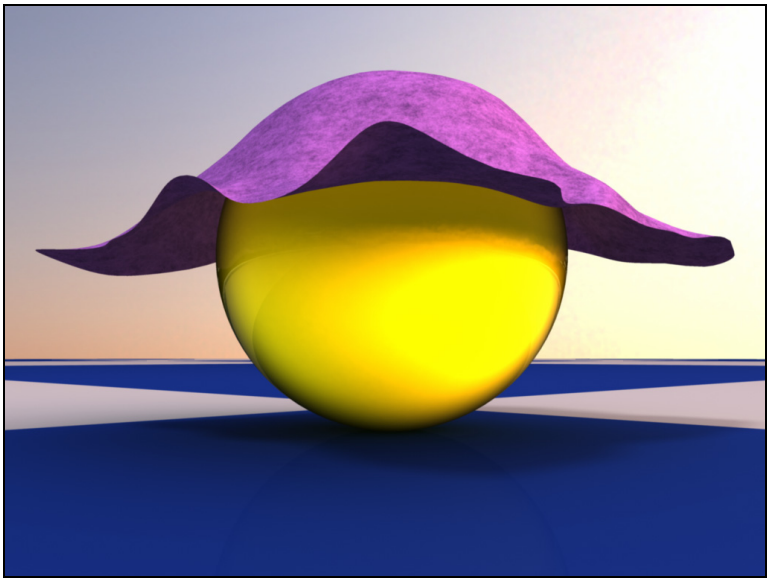

Figure 7. Piece of cloth falling over a sphere

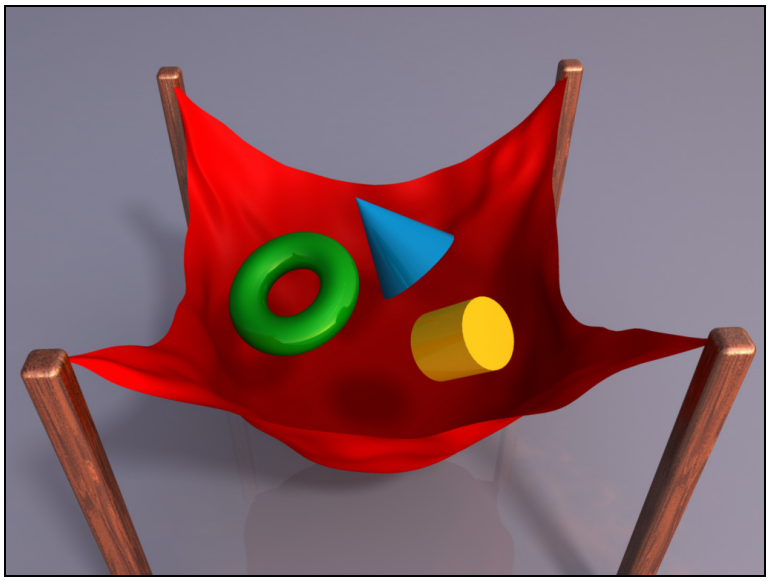

Figure 8. Different objects falling into a piece of cloth

5. CONCLUSION

The presented impulse-based methods for cloth simulation allow the accurate handling of inextensible textiles. These textiles are simulated by using a mesh of particles linked by distance constraints which are satisfied by the computation of impulses. An impulse is determined by using a preview of the corresponding constraint state. The constraint dependencies in a mesh are resolved either iteratively or by solving a system of linear equations for the correction impulses. The extensibility of the cloth can be controlled by a tolerance value. Totally inextensible textiles can be simulated with the introduced SLE method.

One of the main features of the new method is its stability. Since a constraint is directly resolved by using a preview, even totally destroyed models can be repaired. Hence, in each step with the iterative method an early result can be obtained at any time by interrupting the iterative process without the loss of stability. This can be used to increase the performance or to meet real-time requirements. Other features are that the method is simple to implement and that collision and contact handling with friction can be easily integrated in the presented simulation process.

REFERENCES

Baraff, David and Witkin, Andrew, 1998. Large steps in cloth simulation. *Computer Graphics 32, Annual Conference Series,* pp. 43–54.

Bayer, Daniel at al., 2009. Impulse-based dynamic simulation on the GPU. *Computer Graphics and Visualization - IADIS Multi Conference on Computer Science and Information Systems,* Algarve, Portugal.

Bender, Jan and Schmitt, Alfred, 2006. Constraint-based collision and contact handling using impulses. *Proceedings of the 19th international conference on computer animation & social agents, Geneva, Switzerland, pp. 3-12.*

Bender, Jan, 2007, Impulse-based dynamic simulation in linear time. *In Journal of Computer Animation and Virtual Worlds*, Vol. 18, No. 4–5, pp 225–233.

Bridson, Robert E. et al., 2002. Robust treatment of collisions, contact and friction for cloth animation. *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques,* New York, NY, USA, pp. 594–603.

Choi, Kwang-Jin and Ko, Hyeong-Seok, 2002. Stable but responsive cloth. *ACM Trans. on Graphics*, Vol. 21, No. 3, pp. 604–611.

Choi, Kwang-Jin and Ko, Hyeong-Seok, 2005. Research problems in clothing simulation. *Computer-Aided Design*, Vol. 37, No. 6, pp. 585–592.

Fuhrmann, Arnulph et al., 2003. Interactive animation of cloth including self collision detection. *WSCG'03*, pp. 141–148.

Georgii, Joachim and Westermann, Rüdiger, 2005. Mass-Spring Systems on the GPU. *Simulation Modelling Practice and Theory, Vol. 13*, pp. 693–702.

Goldenthal, Rony et al., 2007. Efficent Simulation of Inextensible Cloth. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007)*, Vol. 26, No. 3.

Hauth, Michael et al., 2003. Analysis of numerical methods for the simulation of deformable models. *The Visual Computer*, Vol. 19, No. 7-8, pp. 581–600.

House, Donald H. and Breen, David E., 2000. *Cloth modeling and animation*. A.K. Peters, Ltd., Natick, MA, USA.

Magnenat-Thalmann, Nadia and Volino, Pascal, 2005. From early draping to haute couture models: 20 years of research. The *Visual Computer*, Vol. 21, No. 8-10, pp. 506–519.

Provot, Xavier, 1995. Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behavior. *Graphics Interface '95*, Canadian Human-Computer Communications Society, pp. 147–154.

Schmitt, Alfred et al., 2005, On the Convergence and Correctness of Impulse-Based Dynamic Simulation. *Technical Report 2005-17.* University of Karlsruhe, Germany.

Terzopoulos, Demetri et al., 1987. Elastically deformable models. *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques,* New York, NY, USA, pp. 205–214.

Volino, Pascal and Magnenat-Thalmann, Nadia, 2001. Comparing Efficiency of Integration Methods for Cloth Simulation. *CGI '01: Computer Graphics International,* Washington, DC, USA, pp. 265–274.