

Simulation of Tearable Cloth

PIETERJAN BARTELS

A concise overview of interesting techniques.



November 20, 2014

Abstract

In this report, some of the major techniques for Cloth Simulation will be outlined. This includes a mass-spring model, relevant integration methods and collision handling algorithms. Some recent techniques for realistically tearing cloth will also be discussed. The presented techniques will be used in the author's project for Animation Software Engineering.

Keywords: Cloth, Garments, Collision detection, Collision response

1 Introduction

Cloth is ubiquitous in human life. Not only do we as humans (usually) wear garments, realistic environments also require curtains, flags, ... This implies that, if one wants to make a realistic virtual human environment, cloth should be included. This raises the question of what algorithms can be used to simulate cloth and its movement with a computer.

This question has been a topic of a lot of research in the past decades, and is also the topic of this report. However, this report will be focused on describing a set of techniques that will be used in the author's Animation Software Engineering project, rather than on giving a general overview. For a more thorough overview of existing techniques, one can refer to Magnenat-Thalmann and Volino (2005), Choi and Ko (2005), Nealen *et al.* (2006) and (Parent 2012, chap. 7) as starting points.

2 Cloth Model

The first question is which model one will use for the simulation. Existing models can roughly be divided in two categories: particle-based models and finite element methods (Magnenat-Thalmann and Volino 2005). This report will focus on the particle-based approach, which has been used widely (e.g. in (Provot 1995; Bridson *et al.* 2002; Choi and Ko 2002; Müller *et al.* 2007), ...). For an example of a FEM technique, see (Etmuss *et al.* 2003).

In the particle-based approach, the piece of cloth is discretized as a finite amount of point masses, also called particles. The configuration of the point masses tends to differ, but a simple and popular idea is to place them in a regular grid, as proposed by Provot (1995). Other techniques use the vertices of a triangular mesh as particles, e.g. (Selle *et al.* 2009).

In order to simulate the movement of the cloth as a whole, it suffices to simulate the movement of the individual particles. In reality, a particle's movement is determined by the forces working on it, as dictated by Newton's second law:

$$F_i = m_i a_i \quad (1)$$

where m_i is the mass of particle i , a_i is the particle's acceleration, and F_i is the sum of all forces working on the particle. Hence, most models define a set of forces working on the particles, turn those into acceleration, and integrate that into the movement of the particles, which is the subject of the next section.

These forces are usually divided in two groups:

$$F_i = \sum F_{int} + \sum F_{ext}. \quad (2)$$

F_{ext} are any external forces working on the particle (e.g. gravity, drag, wind), and are usually represented in the same way:

$$\begin{aligned} F_{gravity} &= m a_{grav} \\ F_{drag} &= -d_c v^2 \\ F_{wind} &= w_c (n \cdot W) \end{aligned} \quad (3)$$

These all represent a force working on particle: for $F_{gravity}$ we multiply the gravitational acceleration a_{grav} with the mass m of the particle, F_{drag} is computed from the velocity of the particle v and the drag constant d_c and F_{wind} can be found using (the absolute value of) the dot product of the surface normal in the particle n with the direction of the wind W , and multiplying it by a wind constant w_c .

F_{int} are internal forces of the cloth, i.e. forces caused by the relations between different particles. They determine the internal dynamics of the cloth and make it behave realistically. Due to its structure, woven cloth usually resists two types of deformation: in-plane and out-of-plane (Baraff and Witkin 1998; Müller *et al.* 2007). The first type is sometimes

divided in stretch and shear (Provot 1995). In order to be believable, a cloth model should have internal forces that resist both types of deformations.

One of the earliest, but still very widely used ideas, is using springs between the particles to simulate the internal forces. This was first proposed by Provot (1995) and has been used by many (e.g. Bridson *et al.* (2002); Selle *et al.* (2009); Choi and Ko (2002)). The springs used in the model obey Hooke’s law and also have a damper, which is necessary for stability (Baraff and Witkin 1998). Using the notation of (Parent 2012, chap. 7), the spring exerts following forces:

$$\begin{aligned} F_{\text{spring}} &= -k_s(L_c - L_r) \frac{p_2 - p_1}{\|p_2 - p_1\|}, \\ F_{\text{damper}} &= -k_d(\dot{p}_2 - \dot{p}_1) \cdot \left(\frac{p_2 - p_1}{\|p_2 - p_1\|} \right) \left(\frac{p_2 - p_1}{\|p_2 - p_1\|} \right). \end{aligned} \quad (4)$$

Where p_2 and p_1 are the positions of the particles connected to either end of the spring, F_{spring} and F_{damper} are the forces exerted by the spring and its damper respectively, L_c is the current length of the spring (i.e. distance between p_2 and p_1) and L_r is the rest length of the spring. The spring can be controlled with the two constants k_s , the spring constant, and k_d , the damper constant.

While being popular for its simplicity, this mass-spring is not physically accurate. A better representation is to define energy functions based on the deformation of the cloth elements (e.g. triangles). The resulting force F on a particle at position (x, y, z) can be computed as:

$$F = \left(\frac{\partial E(S)}{\partial x}, \frac{\partial E(S)}{\partial y}, \frac{\partial E(S)}{\partial z} \right). \quad (5)$$

Here $E(S)$ is the mentioned energy function, a function of S , a variable representing the complete state of the cloth. Examples of techniques using this type of internal forces are (Baraff and Witkin 1998; Choi and Ko 2002; Bridson *et al.* 2003).

3 Integration

Given the forces on the particles and the very well-known relation between acceleration, velocity and position, we can write the problem of finding the movement of said particles as a differential equation (notation from (Baraff and Witkin 1998)):

$$\frac{d^2\mathbf{x}}{dt^2} = \mathbf{M}^{-1}\mathbf{F}. \quad (6)$$

Where \mathbf{x} is a vector containing the geometric positions of all particles in the system, the matrix \mathbf{M} represents the mass distribution of the system and \mathbf{F} is a vector containing the resulting force on every particle.

It is impossible to solve equation (6) analytically, and all particle-based techniques for cloth simulation use some numerical integration scheme on discretized timesteps to solve it (Provot 1995; Bridson *et al.* 2002; Selle *et al.* 2009). All of these integration schemes try to find an estimate for the state of the system (i.e. the piece of cloth) at time step i based on timestep $i - 1$.

The easiest of these schemes are straightforward explicit integration, where every new variable for timestep i is computed based on variables of timestep $i - 1$ only. The simplest example is the explicit euler integration scheme (Provot 1995):

$$\begin{aligned} \mathbf{x}_i &= \mathbf{x}_{i-1} + h\mathbf{v}_{i-1}, \\ \mathbf{v}_i &= \mathbf{v}_{i-1} + h\mathbf{M}^{-1}\mathbf{f}_{i-1}. \end{aligned} \quad (7)$$

\mathbf{x}_i is the vector of positions at timestep i and \mathbf{v}_i is a similar vector with the velocities of the particles. Furthermore, h is the used timestep, and \mathbf{M} is still the mass distribution matrix. It is immediately visible that all

variables in timestep i only depend on timestep $i - 1$, and evaluating can be done per particle (Parent 2012, chap. 7). Therefore, implementing this scheme (and similar explicit schemes) is fairly straightforward.

However, Baraff and Witkin (1998) have shown explicit schemes to be unstable when solving differential equations of the form (6), especially when the equation is stiff. For cloth simulation, the stiffness of the differential equation is mainly coming from high in-plane deformation resistance (Baraff and Witkin 1998). Unfortunately, this resistance should be high to avoid super-elasticity (Provot 1995).

This instability makes that explicit schemes are bound to small timesteps, leading to bad performance. Baraff and Witkin (1998) used an implicit integration scheme to avoid this problem and take large timesteps. As an example, we take a look at the backward euler integration scheme:

$$\begin{aligned}\mathbf{x}_i &= \mathbf{x}_{i-1} + h\mathbf{v}_i \\ \mathbf{v}_i &= \mathbf{v}_{i-1} + h\mathbf{M}^{-1}\mathbf{F}_i\end{aligned}\tag{8}$$

The symbols are the same as in equation (7). In fact this equation and equation (7) are very similar. The important difference is that \mathbf{v}_i now depends on \mathbf{F}_i , the forces in timestep i , which are in turn a direct result of \mathbf{x}_i and \mathbf{v}_i (which are unknown). As a result, equation (8) cannot simply be filled in but has to be solved as a system of equations, in contrast to equation (7).

Baraff and Witkin (1998) apply a Taylor series expansion to \mathbf{F}_i , in order to turn equation (8) into a linear equation:

$$\mathbf{F}_i = \mathbf{F}_{i-1} + \frac{\partial\mathbf{F}}{\partial\mathbf{x}}\Delta\mathbf{x} + \frac{\partial\mathbf{F}}{\partial\mathbf{v}}\Delta\mathbf{v}.\tag{9}$$

With $\Delta\mathbf{v} = \mathbf{v}_i - \mathbf{v}_{i-1}$ and $\Delta\mathbf{x} = \mathbf{x}_i - \mathbf{x}_{i-1}$.

By taking and reorganizing the bottom of equation (8), filling in the

given approximation of \mathbf{F}_i , Baraff and Witkin (1998) find following linear system of equations to solve for $\Delta\mathbf{v}$:

$$\left(\mathbf{I} - h\mathbf{M}^{-1}\frac{\partial\mathbf{F}}{\partial\mathbf{v}} - h^2\mathbf{M}^{-1}\frac{\partial\mathbf{F}}{\partial\mathbf{x}}\right)\Delta\mathbf{v} = h\mathbf{M}^{-1}\left(\mathbf{F}_{i-1} + h\frac{\partial\mathbf{F}}{\partial\mathbf{x}}\mathbf{v}_{i-1}\right). \quad (10)$$

Here \mathbf{I} is the only new symbol, being the identity matrix. Note that we now need to be able to differentiate the forces in the cloth model to use this integration scheme. Other examples of authors using similar (semi-)implicit integration are Choi and Ko (2002); Bridson *et al.* (2003); Selle *et al.* (2009). Semi-implicit integration can be used to avoid the artificial damping present in the work of Baraff and Witkin (1998) (Choi and Ko 2002; Bridson *et al.* 2003). Most techniques (e.g. Baraff and Witkin (1998); Bridson *et al.* (2003)) solve this system using a variant of the conjugate gradient method (Trefethen and Bau 1997, chap. 38).

Since implicit integration can be tricky to implement, it is often avoided. Systems using explicit integration use a less stiff differential equation (i.e. lower deformation resistance), and avoid super-elasticity another way. One such way is adding constraints to the system, e.g. a constraint specifying a maximum distance between two particles. These constraints work directly on the particle positions and are usually satisfied in an iterative Gauss-Seidel fashion, after the integration. This iteratively satisfying different constraints converges to a solution with all constraints satisfied, even though the constraints are satisfied independently (Parent 2012; Müller *et al.* 2007, chap. 7).

This idea of constraints is used in many systems, for varying reasons. Provot (1995) used it solely for avoiding super-elasticity. Even Baraff and Witkin (1998) use it, despite their implicit integration: they use it for collision response. Müller *et al.* (2007), inspired by Jakobsen (2001), on the other hand use the constraint system for almost everything, going as far as replacing the springs by constraints. This results in a system where integration is only needed for the external forces, and this allows them to use a simpler integration scheme (i.e. forward Euler).

4 Collisions

In order to have a physically plausible cloth simulation, we need some way of avoiding intersection between the cloth and other objects, and between different parts of the cloth itself. A lot of research has been performed in this topic, but due to the limited amount of time available for the project, the report will not go into very much detail here.

Accurate collision handling has two main problems: collision detection and collision handling. For the first most techniques usually only consider triangles, since most objects in computer graphics are built using triangles. A widely used method for detecting triangle-triangle intersection is the cubic solver approach, as introduced by Provot (1997). It involves finding the roots of a cubic equation for every point-triangle and edge-edge pair. Recently, another interesting technique was presented by Brochu *et al.* (2012), which reduced triangle-triangle intersection to a series of ray-bilinear patch intersection tests. Whatever the triangle-triangle method, all detection techniques use forms of intersection test culling (i.e. bounding volume hierarchies and angular culling techniques (Provot 1997)).

Several techniques exist for the latter, collision handling. A very elegant solution was proposed by Bridson *et al.* (2002). Rather than going for one technique, the authors combine repulsion techniques, impulse-based changes and rigid impact zones into one robust algorithm. In (Selle *et al.* 2009), details about the method from (Bridson *et al.* 2002) are changed to improve performance, and make the technique tractable for high-resolution pieces of cloth.

While these techniques are all interesting in their own right, for this project the author would like to go more into details of other areas of cloth simulation, such as tearing. For this reason, it is deemed appropriate to implement collision handling based on iterative constraint solving

as described by Müller *et al.* (2007), which is easier to implement and leaves more time for other techniques.

5 Tearing

A very interesting part of the behaviour of cloth is the way it tears. This behaviour of cloth has been studied far less than the techniques described in the previous sections, although some publications exist or make mention of an ad-hoc method use to enable tearing (Molino *et al.* 2004; Müller *et al.* 2007; Metaaphanon *et al.* 2009; Souza *et al.* 2014).

Most methods (Molino *et al.* 2004; Müller *et al.* 2007; Souza *et al.* 2014), look for a particle to split, and duplicate that particle, assigning all other elements (springs, constraints, ...) to either the old particle or the duplicate. Figure 1 shows the splitting of a vertex as in (Souza *et al.* 2014), based on a triangular mesh. In this technique, when the strain of some edge is over a user-controlled threshold, one of the vertices of the edge is split. The direction of the edge determines a splitting plane, used to determine whether a triangle should be assigned to the old particle or the new duplicate. This results in a tear in the mesh. Molino *et al.* (2004) and Müller *et al.* (2007) use very similar ideas.

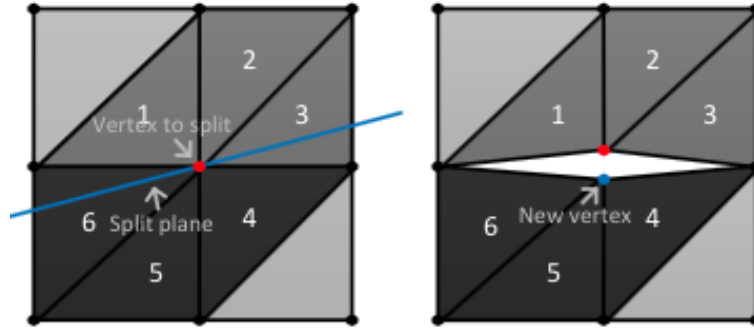


Figure 1: *Tearing of cloth as presented in Souza et al. (2014). The vertex in red is split in two (right). A splitting plane is used to determine how to assign the triangles. Image by Souza et al. (2014).*

However, a different technique is described by Metaaphanon *et al.* (2009). This technique is based on a two-layer model of the cloth. While both layers of the model are based on a similar mass-spring model, the second layer only connects pointmasses in one direction through the cloth, based on the warp/weft pattern of the cloth. The first layer, on the other hand, is a regular grid as in (Provot 1995). The cloth starts out as being completely modelled in the first layer. However, once the strain of a spring becomes too big, it is cut, and all surrounding pointmasses are split into two particles: one for the warp thread, and one for the weft thread. The original pointmass, which was part of the first layer, is replaced by two new particles, part of the second layer. Figure 2 shows the difference between the different layers.

When going from the first layer to the second layer, the position of the two new pointmasses can quite simply be found using the following equations:

$$\begin{aligned} x_{warp} &= x_{base} + \frac{T}{2}n_c, \\ x_{weft} &= x_{base} - \frac{T}{2}n_c. \end{aligned} \tag{11}$$

Where x_{warp} and x_{weft} are the positions of the newly created particles, and x_{base} is the position of the first-layer particle. T is the thickness of the threads, a user-controlled parameter. Finally, n_c is the cloth normal direction in the position of the old particle. The signs of the addition in the formulas may be reversed depending on the weave pattern.

These new pairs of pointmasses are allowed to move relative to each other, albeit constrained, effectively creating yarns of fiber that emerge at the place of the tearing. This results in visible frayed edges in the cloth, which is quite unique compared to other methods (Molino *et al.* 2004; Müller *et al.* 2007; Souza *et al.* 2014). To allow rendering of these frayed edges, the entire piece of cloth is rendered as a collection of Catmull-Rom splines defined by the points of the individual threads. An example result can be found in figure 3.

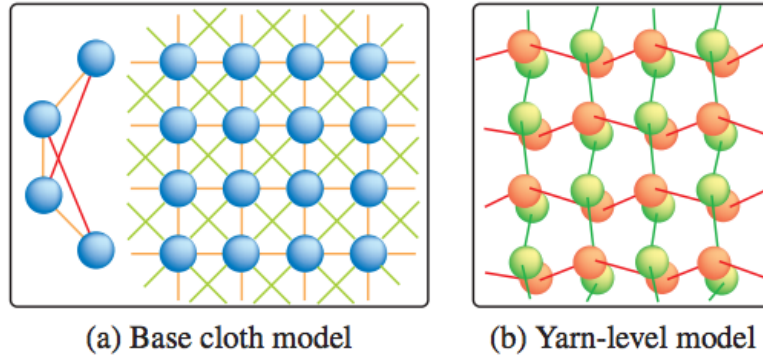


Figure 2: *Visualization of the first (a) and second (b) layer in the two-layer model. When going from the first to the second layer, every particle is split into a weft-particle and a warp-particle. Image by Metaaphanon et al. (2009).*

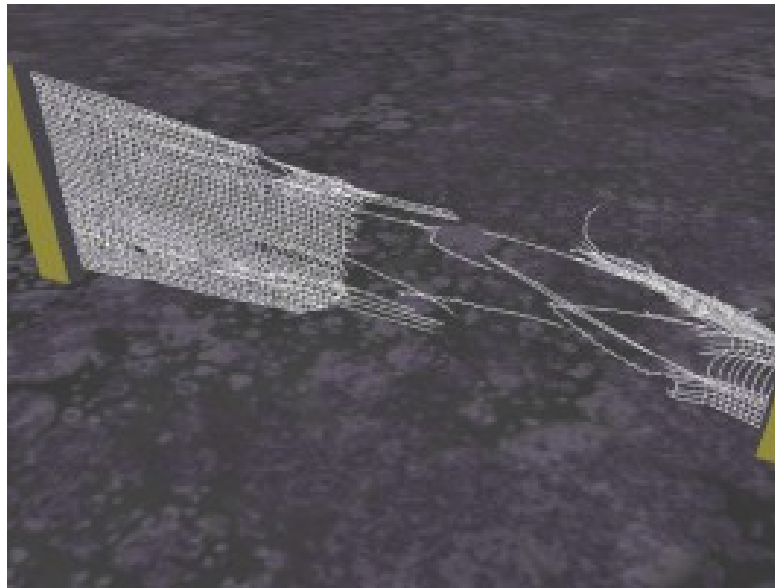


Figure 3: *Example result of the method by Metaaphanon et al. (2009). Image by Metaaphanon et al. (2009).*

6 Conclusion

In this report, some techniques for modelling and tearing a cloth were presented. The author will use these in his project for Animation Software Engineering. To be more specific, the author aims to implement following techniques:

- a particle-based model of cloth, with springs for the internal forces.

Section 2.

- An integration scheme to simulate the movement of said model. Due to their advantages, it would be good to implement a (semi-)implicit scheme, such as the one in Bridson *et al.* (2003). Section 3.
- Some collision handling for believability. To allow more time for implementing integration methods and tearing techniques, the collision handling as in Müller *et al.* (2007) is preferred. Section 4.
- At least one, but preferably two techniques for tearing cloth. By implementing these, the author hopes to gain insight into the challenges particular to tearing, and maybe improve some of the existing techniques. Section 5.

References

- Baraff D. and Witkin A., 1998. Large steps in cloth simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, New York, NY, USA. ACM, 43–54.
- Bridson R., Marino S. and Fedkiw R., 2003. Simulation of clothing with folds and wrinkles. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association, 28–36.
- Bridson R., Fedkiw R. and Anderson J., July 2002. Robust treatment of collisions, contact and friction for cloth animation. *ACM Trans. Graph.*, **21**(3), 594–603.
- Brochu T., Edwards E. and Bridson R., July 2012. Efficient geometrically exact continuous collision detection. *ACM Trans. Graph.*, **31**(4), 96:1–96:7.
- Choi K.-J. and Ko H.-S., 2002. Stable but responsive cloth. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, New York, NY, USA. ACM, 604–611.
- Choi K.-J. and Ko H.-S., 2005. Stable but responsive cloth. In *ACM SIGGRAPH 2005 Courses*, SIGGRAPH '05, New York, NY, USA. ACM.
- Etzmuss O., Gross J. and Strasser W., October/December 2003. Deriving a particle system from continuum mechanics for the animation of deformable objects. **9**(4), 538–550.
- Jakobsen T., 2001. Advanced Character Physics. In *Game Developers Convergence Proceedings*. CMP Media, Inc., 383–401.
- Magenat-Thalmann N. and Volino P., 2005. From early draping to haute couture models: 20 years of research. *The Visual Computer*, **21**(8-10), 506–519.

- Metaaphanon N., Bando Y., Chen B.-Y. and Nishita T., 2009. Simulation of tearing cloth with frayed edges. *Comput. Graph. Forum*, **28**(7), 1837–1844.
- Molino N., Bao Z. and Fedkiw R., August 2004. A virtual node algorithm for changing mesh topology during simulation. *ACM Trans. Graph.*, **23**(3), 385–392.
- Müller M., Heidelberger B., Hennix M. and Ratcliff J., April 2007. Position based dynamics. *J. Vis. Comun. Image Represent.*, **18**(2), 109–118.
- Nealen A., Mller M., Keiser R., Boxerman E. and Carlson M., 2006. Physically based deformable models in computer graphics. *Computer Graphics Forum*, **25**(4), 809–836.
- Parent R., 2012. *Computer Animation: Algorithms and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3 edition.
- Provot X., 1995. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *In Graphics Interface '95*. Canadian Human-Computer Communications society, 147–154.
- Provot X. 1997. 177–189. Collision and self-collision handling in cloth model dedicated to design garments. In Thalmann D. and van de Panne M., editors, *Computer Animation and Simulation 97*, Eurographics, Springer Vienna.
- Selle A., Su J., Irving G. and Fedkiw R., March 2009. Robust high-resolution cloth using parallelism, history-based collisions, and accurate friction. *IEEE Transactions on Visualization and Computer Graphics*, **15**(2), 339–350.
- Souza M., Wangenheim A. and Comunello E., 2014. Fast simulation of cloth tearing. *Journal on Interactive Systems*, **5**, 44–48.
- Trefethen L. N. and Bau D., 1997. *Numerical Linear Algebra*. SIAM.