Cloth Simulation

CGI Techniques Report

TANJA MUNZ

Master of Science
Computer Animation and Visual Effects

Bournemouth
University

21st November, 2014

Abstract

Cloth simulation is a wide and popular area of research. First papers were published in the 1980s and the introduced techniques build the basis for current research. Current publications try to improve computation times and to make the visual results even more realistic. For cloth simulation a variety of different aspects play an important role. These include modelling of cloth, the influence of forces, integration over time and collision. A common and easy way to model cloth is by using the mass-spring system. The influence of forces causes deformation in the shape of the cloth. Numerical integration is used to determine the deformation over time for discrete time steps. Both collision detection and response are necessary for interaction with other objects and with the cloth itself. Each of the mentioned topics will be covered in this report.

Keywords: cloth simulation, mass-spring system, numerical integration, collision detection and response

Contents

1 Introduction

Cloth is a part of daily life and occurs in a variety of different materials and shapes. There are for instance garments, tablecloths, curtains and towels.

All of them can be stretched, bent, skewed, folded and wrinkled. As each can be of a different kind of material with its own distinguishing characteristics like stiffness or weight their behaviour can be diverse in these actions. As a result they form unique folds and buckles.

Cloth simulation is a popular research area in computer animation. It is often needed for computer games and computer animated films. The goal of simulating cloth is to visualise and animate cloth in a realistic way. Additionally, especially the game industry depends on low computation time for real time animation.

A scenario for simulating a piece of cloth could be imagined where cloth is fixed to one or multiple points and is influenced by gravity like a towel hanging on a clothes hook or a tablecloth resting on a table. Another example is cloth colliding with other objects. Furthermore a piece of cloth could be influenced by wind like a flag waving in the breeze.

In these scenarios cloth is under the influence of forces to make it move and to transform its shape. These forces are determined by properties of the cloth and the surrounding environment which includes gravity, wind and collisions.

To determine the simulation over time the cloth has to be modelled and by considering forces its deformation can be calculated.

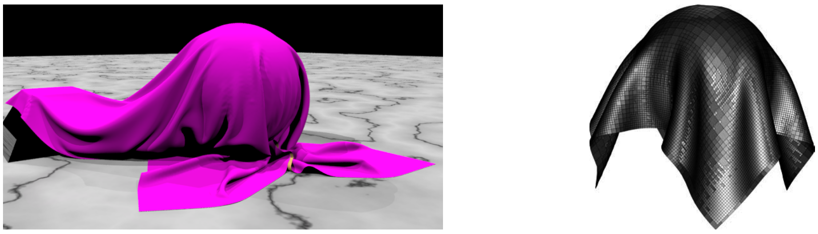
2 Related Work

In the following sections there will be an overview of research in the area of cloth simulation. As there are many publications in this area only a small selection of the most relevant papers for this report are presented. A detailed survey of early research can be found for instance in Ng and Grimsdale (1996). All of the following techniques use physically-based models.

Provot (1995) uses the mass-spring system to model cloth. This model will be explained in-depth in the next chapter. Internal and external forces are applied and the explicit Euler integration is used to determine the deformation of cloth over time. An inconvenient property of this model is the *super-elasticity*. This is visible as a high elongation of springs that occurs for example on cloth influenced by gravity near fixed corners. This problem is solved by adjusting the elongation of springs if their deformation is too large. In Provot (1997) collision handling for this model is explained in detail.

Baraff and Witkin (1998) use an implicit integration method that allows larger time steps compared to earlier techniques and provides stability. Additionally an adaptive time stepping is used. This research uses shear, bend, stretch and damping forces as well as forces due to air-drag, gravity and user-generated forces. Constraints can be set by users or appear between objects and particles. For collision detection a hierarchical bounding box tree is used and for collision response constraints between cloth and an object are enforced or penalty forces are added.

Volino and Thalmann (2000) focus on collision detection and response. Hierarchically organised bounding volumes and a constraint-based collision response scheme are used. For the simulation implicit integration is used.



(a) (b)

Figure 2.1: *Images of simulated cloth: (a) Bridson* et al. *(2002) with collision of cloth with other objects and (b) Villard and Borouchaki (2005) showing adaptive meshing.*

Bridson et al. (2002) concentrate especially on self-collision for detailed time evolving folds and wrinkles. Additionally thickness for cloth is allowed. The previously mentioned *super-elasticity* problem in Provot (1995) is handled by adjusting the velocities instead of positions of mass points. For collision detection an axis-aligned bounding box hierarchy is used. Collision response is handled by repulsion forces that minimize the number of collisions; with further adjustments a fail-safe method is being created.

Choi and Ko (2002) has as main achievement that wrinkles form and disappear in a natural way. An improvement in stability and realism is provided. A semi-implicit integration with large fixed time-steps is used for the simulation.

Villard and Borouchaki (2005) present a method for adaptive meshing for cloth animation by adopting the mass-spring system introduced by Provot (1995) to visualize surfaces in more detail, especially for wrinkles. With this method the number of mesh elements can be reduced and therefore the computation time too.

3 Technical Background

In this chapter the technical background for cloth simulation will be covered.

3.1 Cloth Modelling

A common and easy way to model a piece of cloth is by using the *mass-spring system*. Provot (1995) provides a detailed description; other publications that use this model are for instance by Volino and Thalmann (2000), Choi and Ko (2002), Bridson *et al.* (2002) and Villard and Borouchaki (2005).

In this model cloth is represented by particles (usually also referred to as *mass points*) that are connected by *springs* to their neighbours.
Each *mass point* represents a position in the three dimensional space with a given mass and a velocity.
Springs are the connections of two mass points. The springs are massless, have a rest length (the initial distance of the connected mass points) and properties like stiffness and damping that are crucial for calculating forces.
More formal, a piece of cloth is modelled by $m \times n$ mass points. P_{ij} is a mass point in the grid, where $i = 1, \ldots, m$ and $j = 1, \ldots, n$. Each particle P_{ij} has a given mass m_{ij}, position $\mathbf{x}_{ij}$ and velocity $\mathbf{v}_{ij}$. An illustration for a rectangular piece of cloth is visible in figure 3.1.

Usually there are different types of springs that are connected to different neighbouring mass points. Provot (1995) introduces three kinds of springs:

Structural Springs are oriented along the directions of warps and wefts of cloth. As seen in figure 3.2a they connect horizontally and vertically every neighbouring mass point. They are responsible for stretching along the warps and wefts direction and keeping the structure of the cloth. Supposed there were only these springs a piece of cloth under gravity would collapse to one single line.

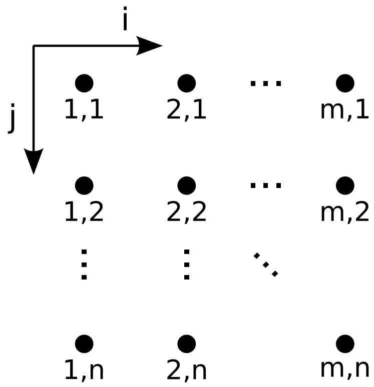


Figure 3.1: *Grid of $m \times n$ mass points with corresponding mass point indices in the grid.*

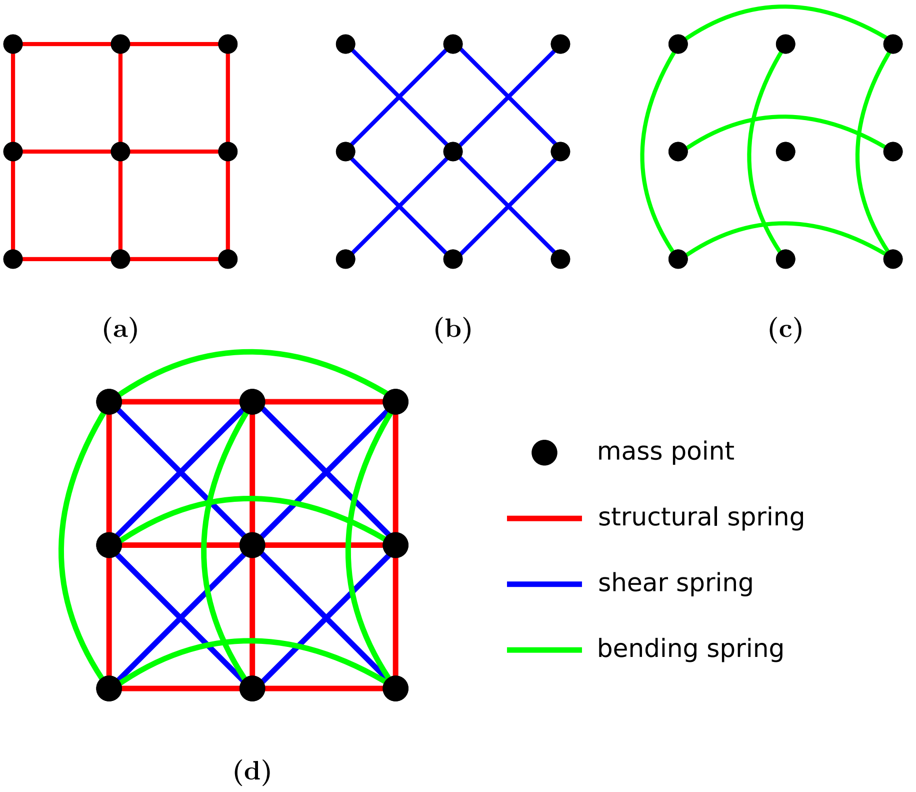


Figure 3.2: *Different types of springs in a mass-spring system: (a) structural springs, (b) shear springs, (c) bending springs and (d) all three types of springs.*

Shear Springs are shown in figure 3.2b. They connect mass points on the diagonal.

Bending Springs can be seen in figure 3.2c They connect particles horizontally and vertically. In contrast to structural springs they connect every second mass point.

All three kinds of springs are visible in figure 3.2d.

3.2 Forces

The mass-spring system has to keep its general shape and should interact with its surrounding environment. For this purpose forces are applied on either the mass points or the springs that pass the forces on to the mass points.

The force on each mass point is based upon Newton's second law of motion: $\mathbf{F}_{ij} = m_{ij} \cdot \mathbf{a}_{ij}$. Here, m_{ij} is the mass of the mass point P_{ij} and the force $\mathbf{F}_{ij}$ causes the acceleration $\mathbf{a}_{ij}$.

There are two different types of forces: internal and external forces. $\mathbf{F}_{ij}$ is the summation of all forces that are applied to P_{ij}.

Internal Forces are responsible for the behaviour of the springs and allow stretching, shearing and bending.

For each mass point P_{ij} that is connected by a spring to the point P_{kl} the following forces can be applied according to Hooke's law.

Stiffness is the tension of the spring P_{ij} linking to its neighbour P_{kl}:

$$\mathbf{F}_s(P_{ij}, P_{kl}) = -k_s \mathbf{x}_{ij,kl}$$

with k_s being the spring constant and $\mathbf{x}_{ij,kl}$ the displacement.
$\mathbf{x}_{ij,kl}$ can be expressed by: $\mathbf{x}_{ij,kl} = \frac{\mathbf{x}_{kl} - \mathbf{x}_{ij}}{|\mathbf{x}_{kl} - \mathbf{x}_{ij}|} \cdot (|\mathbf{x}_{kl} - \mathbf{x}_{ij}| - l_{ij,kl})$;
$l_{ij,kl}$ is the rest length between the points P_{ij} and P_{kl}.

Damping acts against the velocity of the mass points and is used to bring the spring back to its rest length:

$$\mathbf{F}_d(P_{ij}) = -k_d \mathbf{v}_{ij}$$

with k_d being the damping coefficient.

External Forces are a result of the surrounding environment, i.e. by gravity, wind or other objects colliding with the cloth. (For collision handling see chapter 3.4.)

Gravity is determined by

$$\mathbf{F}_g(P_{ij}) = m_{ij} g$$

with $g = 9.81 \frac{m}{s^2}$ being the acceleration of gravity.

The force of *wind* can be calculated by

$$\mathbf{F}_w(P_{ij}) = k_w [\mathbf{n}_{ij} \cdot (\mathbf{u}_w - \mathbf{v}_{ij})] \mathbf{n}_{ij}$$

with k_w being the wind strength, n_{ij} the unit normal of the surface at the mass point, u_w the velocity of wind (Provot 1995).

3.3 Integration

In the following sections there will be a short description of common numerical integration methods for differential equations. Numerical integration is used to determine approximated positions and velocities of the mass points over time. There art two different types of integration: *implicit* and *explicit*.

The usage of different integration methods has a crucial impact on the resulting simulation. Aspects that have to be considered are computation time for one iteration, size of time steps, numerical stability and accuracy.

Time is discretised into time steps of size $\Delta t > 0$. The value for the n-th time step (e.g. for the time $t = n\Delta t$) is y_n.

A comparison of some integration methods used for cloth simulation can be found in Volino and Magnenat-Thalmann (2001).

Explicit Integration

Explicit methods solve differential equations numerically for the next time step by using the result of the current one.

Common used methods are the *Euler method* (used for instance by Provot (1995)) and *Runge-Kutta methods* (mentioned by Baraff and Witkin (1998) and Volino and Magnenat-Thalmann (2001)). Other well known techniques are the *Verlet integration* and the *midpoint method.*

Euler Integration is easy to use. It calculates an approximated value y_{n+1} for the time $t + \Delta t$ by using the value y_n of the previous time step:

$$y_{n+1} = y_n + \Delta t f(t_n, y_n),\ n = 0, 1, \ldots$$

Provot (1995) calculates new positions and velocities of mass points by using a known acceleration. The acceleration $\mathbf{a}_{ij}$ is caused by the force $\mathbf{F}_{ij}(t)$ that is applied to point P_{ij} at any time $t + \Delta t$:

$$\mathbf{a}_{ij}(t + \Delta t) = \tfrac{1}{m_{ij}} \mathbf{F}_{ij}(t)$$

Now, the velocity and position can be calculated accordingly:

$$\mathbf{v}_{ij}(t + \Delta t) = \mathbf{v}_{ij}(t) + \Delta t \mathbf{a}_{ij}(t + \Delta t)$$

$$\mathbf{x}_{ij}(t + \Delta t) = \mathbf{x}_{ij}(t) + \Delta t \mathbf{v}_{ij}(t + \Delta t)$$

This method is inaccurate and not very stable. Furthermore small time steps are needed to prevent numerical instability.

A more accurate solution can be determined by using for example Runge-Kutta integration.

Runge-Kutta Integration While the Euler integration only uses the value of the function $f(t, y)$ at the beginning of the interval the Runge-Kutta integration uses several approximated values at times within the interval.

Here are the equations for Runge-Kutta fourth-order integration that is most often used:

$$y_{n+1} = y_n + \tfrac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4),\ n = 0, 1, \ldots$$

with

$$k_1 = f(t_n, y_n)$$
$$k_2 = f(t_n + \tfrac{\Delta t}{2}, y_n + \tfrac{1}{2}k_1\Delta t)$$
$$k_3 = f(t_n + \tfrac{\Delta t}{2}, y_n + \tfrac{1}{2}k_2\Delta t)$$
$$k_4 = f(t_n + \Delta t, y_n + k_3\Delta t)$$

Implicit Integration

Implicit methods are used for instance in Baraff and Witkin (1998) and Volino and Thalmann (2000).

The implicit Euler integration can be written as follows:

$$y_{n+1} = y_n + \Delta t f(t_{n+1}, y_{n+1}),\ \ n = 0, 1, \ldots$$

Here, $f(t_{n+1}, y_{n+1})$ is not explicitly given as it contains a condition of the time step itself. To solve this nonlinear equation approximation methods can be used.

This method might need more time per iteration than the explicit methods but larger time steps can be used. Therefore it can be faster. Additionally it might have benefits in stability and accuracy.

3.4 Collision

It is not possible that two objects share the same space at the same time. Therefore collision has to be detected and intersections have to be avoided. This is according to Provot (1997) the most time-consuming part in cloth simulation. Generally there are two problems: collision detection and collision response.

Collision Detection

Collision detection has the goal to detect the geometrical position of contacts or intersections. In cloth simulation attention should be paid to two groups of collision: collision between cloth and another object and self-collision.

An easy way to implement collision detection is by comparing each polygon of the simulation with each other. This brute-force method requires very much computation time. There are many ways to optimize the detection.

A common method used in cloth simulation is to organize surfaces in a hierarchy of bounding volumes (e.g. used by Bridson *et al.* (2002) and Volino and Thalmann (2000)).
Bridson *et al.* (2002) use axis-aligned bounding boxes that are organized hierarchically. The hierarchy is built starting by single triangles of the surface and then pairing off adjacent triangles to create parent nodes. The triangles are the leaf nodes and finally there is one root node that contains all triangles. The dimensions of the bounding boxes for the leaf nodes are boxes around the triangles and for the parents the union of the extents in each axis direction of the bounding boxes of the children. The structure of the hierarchy has to be determined only once but the bounding volumes have to be recomputed for each time step.
To determine collision first the bounding boxes of the root nodes are used. If there is a collision children are checked recursively until no collision is detected or the leaf nodes are reached. Then the points of affected triangles can be checked against faces and edges against other edges.

In the case of self-collision Provot (1997) outlines that areas of cloth with "low curvature" are not able to self-intersect and the number of self-collision tests can be reduced.

An important aspect due to time discretisation that should not be neglected is that polygons can move through each other between two time steps without colliding.

Collision Response

After detecting a collision the system has to react to avoid intersections. There are two main ways to handle collision: Forces can be added and handled in the next time step of the integration or the particle positions and velocities can be directly changed.

The first method is often used for self-collisions. For instance in Baraff and Witkin (1998) penalty forces are added to push the cloth apart from each other.

The second one is usually used for collisions with other objects as Baraff and Witkin (1998) enforces constraints for collisions between cloth and solids.

Due to new collisions that could arise from the collision handling the process might be repeated several times.

4 Summary

This report provides an overview of research in the area of cloth simulation. The mass-spring system is described as a common and easy method to model cloth by using mass points that are connected by springs. Furthermore different kinds of forces that can be applied to cloth are introduced. When choosing an integration method for cloth simulation over time the computation time and the quality of the resulting simulation has to be considered. Additionally, it is important to choose adequate time steps. Finally, the methods for collision detection and response are outlined.

Bibliography

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., Fedkiw R. and Anderson J., July 2002. Robust Treatment of Collisions, Contact and Friction for Cloth Animation. *ACM Trans. Graph.*, **21**(3), 594–603.

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.

Ng H. and Grimsdale R., Sep 1996. Computer Graphics Techniques for Modeling Cloth. *Computer Graphics and Applications, IEEE*, **16**(5), 28–41.

Provot X., 1995. Deformation Constraints in a Mass-Spring Model to Describe Rigid Cloth Behaviour. In *Graphics Interface*. Canadian Information Processing Society, 147–147.

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.

Villard J. and Borouchaki H., 2005. Adaptive Meshing for Cloth Animation. *Engineering with Computers*, **20**(4), 333–341.

Volino P. and Magnenat-Thalmann N., 2001. Comparing Efficiency of Integration Methods for Cloth Simulation. In *Computer Graphics International 2001. Proceedings*, 265–272.

Volino P. and Thalmann N., 2000. Implementing Fast Cloth Simulation with Collision Response. In *Computer Graphics International, 2000. Proceedings*, 257–266.