계산량 조정이 가능한 실시간 옷감 시뮬레이션 방법

김 명 준†

요 약

옷감 시뮬레이션에서 explicit 방법은 시간격(time step)이나 스프링 인장력이 큰 경우 안정적이지 않아 사용할 수 없기 때문에, 느리지만 안정성이 높은 implicit 방법이 사용되어 왔다. 시간격을 크게 하는 것은 시뮬레이션을 빠르게 진행시키는 효과가 있고, 큰 인장력의 스프링으로는 잘 늘어나지 않는 옷감을 표현할 수 있다.

다른 방법으로는 explicit 방법처럼 계산이 빠르면서도 implicit 방법처럼 안정적인 특성을 가지는 semi-implicit 방법들을 찾는 연구가 진행되어 왔다. 본 논문에서는 Kang(Kang and Cho 2002)의 방법을 개선하여 거의 explicit 방법에서 완전한 implicit 방법까지 조절이 가능한 실시간 옷감 시뮬레이션 방법을 제안한다. 이 방법은 explicit 방법처럼 간단하고 빠르면서 implicit 방법처럼 시간격(time step)이나 스프링 인장력이 큰 경우에도 안정적으로 작동한다. 또한 기존의 semi-implicit 방법과 비교해서 비교적 작은 인공적 감쇠(artificial damping) 현상을 가진다.

키워드 : 옷감 시뮬레이션, Explicit 방법, Implicit 방법, 안정성

A Scalable Semi-Implicit Method for Realtime Cloth Simulation

Myoung-Jun Kim†

ABSTRACT

Since well-known explicit methods for cloth simulation were regarded unstable for large time steps or stiff springs, implicit methods have been proposed to achieve the stability. Large time step makes the simulation fast, and large stiffness enables a less elastic cloth property.

Also, there have been efforts to devise so-called semi-implicit methods to achieve the stability and the speed together. In this paper we improve Kang's method (Kang and Cho 2002), and thus devise a scalable method for cloth simulation that varies from an almost explicit to a full implicit method. It is almost as fast as explicit methods and, more importantly, almost as stable as implicit methods allowing large time steps and stiff springs. Furthermore, it has a less artificial damping than the previously proposed semi-implicit methods.

Key Words : Cloth Simulation, Explicit Method, Implicit Method, Stable Simulation

1. Introduction

Since Terzopoulos et al.[18] correctly characterized cloth simulation as deformable surfaces and other research groups [5, 3, 19, 13, 11, 8, 14] dealt with the problem, cloth simulation has been regarded as one of the most important problems in computer animation. Cloth dynamics can be expressed in a set of differential equations (usually mass-spring model), which are called governing equations. The physics simulation for a computer animation is somewhat different from the one for other purpose like mechanical engineering. The accuracy of the simulation is not very important for animation purpose; we need an accuracy to the extent which pleases people with visually plausible physics. Especially for a real-time application, we need a fast simulation method even though it sacrifices some degree of accuracy. Also, the stability of the method is a must-have because that allows a broad range of control parameters, beneficial in controlling animation [1].

Explicit methods in [17] are simple and fast, however they are known to be unstable for large time steps or stiff springs(i.e. stiff material). A small time step is required to make explicit schemes stable, and thus resulting in slow simulation. Stability can be greatly improved by using implicit methods [1, 6]. Implicit methods require solving a linear system that takes considerable computation time, as the geometric complexity of the cloth increases. It makes implicit methods much slower than explicit methods. On the other hand a large time step can be used to speed up the simulation; however, there is a limit in increasing the time step for the simulation to describe a natural phenomenon. Moreover, a larger time step causes higher computation (i.e. more iterations) in solving the linear system.

† 정 회 원 : 이화여자대학교 디지털미디어학부 조교수
논문접수 : 2005년 9월 30일, 심사완료 : 2006년 2월 24일

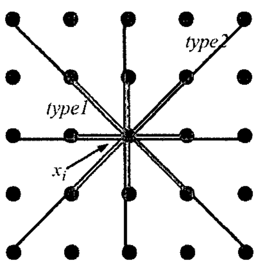
There is another class of approaches called semi-implicit methods[9, 16, 15, 13], a compromise between explicit and implicit methods using a very approximated solving. This possibly results in a stability of an implicit method and also a speed comparable to that of an explicit method. While those methods are fast and stable, they introduce unnecessary artificial damping. The unnecessary damping causes problems of losing momentum and resisting forming wrinkles, thus making cloth unrealistic. To heal this, Desbrun et al.[9] proposed a post-correction of angular momentum as a compensation for the momentum loss. Choi et al.[6] noted that an excessive artificial damping makes the cloth motion unresponsive-slower motion and lesser wrinkles than expected-and that responsiveness is also an important factor to the fidelity of the simulation. On the other hand, there is also an approach of mixing the efficiency of the explicit method and the stability of the implicit method in [4]. They used several steps of explicit/implicit steps but implicit steps are only for integrating the damping and explicit steps are responsible for integrating forces, which result in a limited improvement in the stability over explicit methods.

Other various techniques have been proposed to improve the speed of cloth simulation. A multi-resolution approach is applied in [12], where computationally expensive implicit solving step is performed in coarser mesh. The motion of tight clothes is statically dependent on the body posture, thus may not need a full dynamics simulation. Cordier et al.[7] proposed a real-time simulation technique based on this. Gershbein et al.[10] also used this idea to create an artistic effect in cloth motion. There is also a data driven approach utilizing pre-computed cloth motion[8].

This paper proposes a semi-implicit method for a real-time cloth simulation, which is fast, stable, and has less artificial damping. It does not need to solve a massive linear system, so it is very fast like an explicit method. The remainder of the paper is organized as follows. In Section 2, we first state the mass-spring model that we are using throughout the paper, and review explicit and implicit methods. We present our method and describe its property in Section 3. We also compare with previous semi-implicit methods in Section 4. Finally, we present experimental results in Section 5, and discuss our conclusions in Section 6.

2. Mass-spring model for cloth simulation

We choose a simple mass-spring model for cloth simulation; it is most popular, easy to implement, and computationally light. Also, it provides a flexibility of mixing strong and weak springs, which enables us to configure a cloth bending softly but stretching hardly. We use a quadrilateral mesh of mass-spring used in [6]. Each mass point is linked to the adjacent mass points by strong *type1* springs, and also to the next adjacent nodes by weak *type2* springs, as shown in (Fig. 1).



(Fig. 1) Mass-spring interconnections.

〈Table 1〉 Summary of Notations. Boldface with subscripts indicates a 3-dimensional vector value. Boldface without subscripts indicates overall states containing all the subscripted variables.

Notation	Description
$\mathbf{x}_i, \mathbf{v}_i$	Position and velocity of mass point i
$\mathbf{x} = (\mathbf{x}_i)$ $\mathbf{v} = (\mathbf{v}_i)$	Overall position & velocity state of the system
$\mathbf{F}_i$	Total force acting on mass point i
$\mathbf{F} = (\mathbf{F}_i)$	Overall force state of the system
$\mathbf{x}_{ij}$	Relative position of mass i respect to j, i.e. $\mathbf{x}_j - \mathbf{x}_i$
L_{ij}	Rest-length of the spring between mass i and j
$\mathbf{f}_{ij}$	Spring force acting on mass i induced by mass j
$\partial \mathbf{f}_{ij}/\partial \mathbf{x}_{ij}$	Jacobian matrix of $\mathbf{f}_{ij}$ respect to $\mathbf{x}_{ij}$
$\partial \mathbf{F}_i/\partial \mathbf{x}_i$	Jacobian matrix of $\mathbf{F}_i$ respect to $\mathbf{x}_i$
k_{ij}	Stiffness of spring connecting mass i and j
Δt	Time step
m	Mass of each mass point

For the sake of simplicity, we assume that all the mass points have the same mass m. A mass point i is linked to other points j with linear springs of rest-length L_{ij} (length in resting state) and stiffness k_{ij}. The position and the velocity of mass point i are denoted by $\mathbf{x}_i$ and $\mathbf{v}_i$, respectively. And, $\mathbf{F}_i$ denotes the total spring force acting on a mass point i. Superscript '$*$' indicates the time after Δt. For example, $\mathbf{x}_i^* = \mathbf{x}_i + \mathbf{v}_i \Delta t$. Refer to Table 1 for a summary of the notations used in this paper.

2.1 Forces and their derivatives

Force vector $\mathbf{f}_{ij}$ acting on node i, induced by the spring between mass points i and j, can be expressed as

$$\mathbf{f}_{ij} = k_{ij}\left(1 - \frac{L_{ij}}{|\mathbf{x}_{ij}|}\right)\mathbf{x}_{ij}\ .$$

The Jacobian matrix of the force vector is then

$$\frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{x}_{ij}} = k_{ij}\left(1 - \frac{L_{ij}}{|\mathbf{x}_{ij}|}\right) + k_{ij}\frac{L_{ij}}{|\mathbf{x}_{ij}|^3}\mathbf{x}_{ij}\mathbf{x}_{ij}^T . \qquad (1)$$

The Jacobian $\partial \mathbf{f}_{ij}/\partial \mathbf{x}_{ij}$ is equal to the spring stiffness k_{ij} along the direction of the spring. In the out-of-plane direction (perpendicular direction to springs), the second term of $\partial \mathbf{f}_{ij}/\partial \mathbf{x}_{ij}$ vanishes, i.e., the Jacobian becomes $k_{ij}(1 - L_{ij}/|\mathbf{x}_{ij}|)$, which is small normally when the spring is maintained to have a small deformation ($|\mathbf{x}_{ij}| \approx L_{ij}|$). This property of $\partial \mathbf{f}_{ij}/\partial \mathbf{x}_{ij}$ has an important role in stabilizing the system in implicit methods; large value of $\partial \mathbf{f}_{ij}/\partial \mathbf{x}_{ij}$ introduces a damping that smoothly filters the velocity and makes the system stable. This damping also causes an inaccurate simulation. However, the damping is minimized in the out-of-plane direction due to the smaller value of $\partial \mathbf{f}_{ij}/\partial \mathbf{x}_{ij}$; unnecessary damping is avoided.

The total spring force acting on mass point i is $\mathbf{F}_i = \sum_j \mathbf{f}_{ij}$. Throughout this paper, the summation $\sum_j$ denotes the sum for all the mass points j linked to the mass point i. External forces such as gravity can be added to $\mathbf{F}_i$, though the Jacobian $\partial \mathbf{F}_i/\partial \mathbf{x}_i$ doesn't need to be changed, assuming the external force is independent of $\mathbf{x}_i$.

2.2 Explicit and implicit methods

The following explicit method, called *symplectic Euler*, can be used to simulate a mass-spring model of cloth [17, 1]:

$$\begin{aligned} \mathbf{v}^* &= \mathbf{v} + \mathbf{F}\tfrac{\Delta t}{m} \\ \mathbf{x}^* &= \mathbf{x} + \mathbf{v}^*\Delta t \end{aligned} \qquad (2)$$

The symplectic Euler method (Eq. (2)) becomes unstable for large time steps or stiff springs.

The stability is inversely proportional to $k\Delta t^2/m$ (k is the average stiffness of springs), which means we cannot use large values for Δt and k. We do not need to force unnecessarily large time steps even for an animation purpose due to non-linearity of spring forces. However, it is desirable to have stiff *type1* springs. Small stiffness keeps the simulation stable, but it yields a rubber-like cloth motion. Since clothes resist from stretching, a larger stiffness is required to simulate natural looking cloth. To allow large stiffness, we should reduce the time step for a stable simulation with the explicit method. Notice that stability is improved quadratically as Δt decreases.

The stability can be improved greatly by the following implicit Euler method:

$$\begin{aligned} \mathbf{v}^* &= \mathbf{v} + \mathbf{F}^*\tfrac{\Delta t}{m} \\ \mathbf{x}^* &= \mathbf{x} + \mathbf{v}^*\Delta t \end{aligned} \qquad (3)$$

Since $\mathbf{F}^*$(the force in the next time step) is unknown, the above equations should be solved implicitly. This implicit scheme is stable for a very large $k\Delta t^2/m$ because the solver finds a stabilized $\mathbf{F}^*$. Since Eq. (3) is non-linear, usually a linearization ($\mathbf{F}^* = \frac{\partial \mathbf{F}}{\partial \mathbf{x}}\Delta \mathbf{x} = \frac{\partial \mathbf{F}}{\partial \mathbf{x}}\mathbf{v}^*\Delta t$) is used:

$$\mathbf{v}^* = \mathbf{v} + (\mathbf{F} + \frac{\partial \mathbf{F}}{\partial \mathbf{x}}\mathbf{v}^*\Delta t)\frac{\Delta t}{m} . \qquad (4)$$

Finally, solving for the unknown $\mathbf{v}^*$ yields the following linear system:

$$(\mathbf{I} - \frac{\Delta t^2}{m}\frac{\partial \mathbf{F}}{\partial \mathbf{x}})\mathbf{v}^* = \mathbf{v} + \mathbf{F}\frac{\Delta t}{m} . \qquad (5)$$

The explicit methods (Eq. (2)) and the implicit method (Eq. (5) have the same form except the system matrix $\mathbf{A} = \mathbf{I} - \frac{\Delta t^2}{m}\frac{\partial \mathbf{F}}{\partial \mathbf{x}}$. The inverse matrix $\mathbf{A}^{-1}$ acts like an smoothing filter on velocities, producing an *artificial viscosity*. This artificial viscosity stabilizes the simulation. However, it also causes an inaccurate simulation resulting in a damped motion. Fortunately, this fictitious damping occurs along the direction with a tension, which means no damping occurs in the direction perpendicular to springs. Since the mass-spring model is designed to have less stiffness for bending (see (Fig. 1), the damping for bending motion will be minimal.

Baraff et al.[1] presented a flagship work of using implicit method in the field of cloth animation. Choi et al.[6] improved responsiveness of the implicit cloth simulation by using asymmetric force function for the *type1* springs as follows:

$$\mathbf{f}_{ij} = \begin{cases} k_{ij}\left(1 - \frac{L_{ij}}{|\mathbf{x}_{ij}|}\right)\mathbf{x}_{ij}, & \text{if } |\mathbf{x}_{ij}| > L_{ij} \\ 0, & \text{otherwise} \end{cases} \qquad (6)$$

$$\frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{x}_{ij}} = \begin{cases} k_{ij}\left(1 - \frac{L_{ij}}{|\mathbf{x}_{ij}|}\right) + k_{ij}\frac{L_{ij}}{|\mathbf{x}_{ij}|^3}\mathbf{x}_{ij}\mathbf{x}_{ij}^T, & \text{if } |\mathbf{x}_{ij}| > L_{ij} \\ 0, & \text{otherwise} \end{cases} \qquad (7)$$

For the compressing motion, only *type2* spring acts and the repulsive force becomes small. The stiffness $\partial \mathbf{F}/\partial \mathbf{x}$ is also small and $\mathbf{A} \approx \mathbf{I}$, thus solving Eq. (5) results in a reduced artificial viscosity for the compressing motion.

This eliminates a resistance to starting wrinkles.

3. A scalable semi-implicit method

In this section, we present a scalable semi-implicit method for cloth simulation that varies from an almost explicit to a full implicit scheme. It is almost as fast as explicit schemes, as stable as implicit schemes. Also it has relatively less artificial damping. The implicit equation (Eq. (14)) is rewritten for individual mass point i as follows:

$$\mathbf{v}_i^* = \mathbf{v}_i + \mathbf{F}_i \frac{\Delta t}{m} + \sum_i \frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{x}_{ij}} (\mathbf{v}_j^* - \mathbf{v}_i^*) \frac{\Delta t^2}{m},$$

where $\sum_j$ is the summation for all mass point j linked to i. We solve the above implicit equation for the diagonal terms $\mathbf{v}_i$*. Since $\mathbf{I} + \frac{\Delta t^2}{m} \sum_j \partial \mathbf{f}_{ij} / \partial \mathbf{x}_{ij}$ is a 3×3 matrix, it can be easily inverted so that we have an explicit form:

$$\mathbf{v}_i^* = \left(\mathbf{I} + \frac{\Delta t^2}{m} \sum_j \frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{x}_{ij}} \right)^{-1} \left(\mathbf{v}_i + \mathbf{F}_i \frac{\Delta t}{m} + \sum_j \frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{x}_{ij}} \mathbf{v}_j^* \frac{\Delta t^2}{m} \right). \quad (8)$$

The above equation casts $\mathbf{v}_i$* in a recurrence relation in terms of $\{\mathbf{v}_j{*}\}$ on the righthand side. We initially set the unknown $\mathbf{v}_i$* to the current $\mathbf{v}_i$, and iterate Eq. (8) several times to get the final $\mathbf{v}_i$*. It is very important to update $\mathbf{v}_i$* in place and make the updated value available immediately for other updates; If we update $\mathbf{v}_i$* after the whole iterations completed using temporary variables for $\mathbf{v}_i$*, the simulation becomes unstable. This is a 3×3 block *Gauss-Seidel* iteration for solving Eq. (5). We iterate it for a fixed number of times η. Even for a small η, we found that it is very stable and generates yet plausible cloth motion. We make use of this η as a design parameter to balance between simulation speed and correctness. Since all the terms in Eq. (8) except $\mathbf{v}_i$* remain constant during the iteration, so the iteration can be performed very quickly by building lookup tables for the constant terms. This procedure is summarized as follows:

```
Procedure: Scalable-Method($\eta$)
  build tables for $(\mathbf{I}+\frac{\Delta t^2}{m}\sum_j \frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{x}_{ij}})^{-1}$, $\mathbf{F}_i$, $\frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{x}_{ij}}$
  initialize $\mathbf{v}_i^* = \mathbf{v}_i$
  iterate $\eta$-times
     update all $\mathbf{v}_i$* in place by Eq.  (8)
```

We also use the asymmetric spring model (Eq. (6)) proposed by Choi et al.[6] for *type1* springs. This ensures that $\mathbf{I} + \frac{\Delta t^2}{m} \sum_j \partial \mathbf{f}_{ij} / \partial \mathbf{x}_{ij}$ is positive definite thus invertible. Surprisingly, the stability is degraded in our method if we change the spring model from the symmetric one (the same spring constant for both of stretching and compressing) to the Choi's model (no force for compressing). We found that this is because $\partial \mathbf{f}_{ij} / \partial \mathbf{x}_{ij}$ of Choi's spring model has a discontinuity. Interestingly, this discontinuity did not cause the problem in the implicit methods, because the mutually connected springs are counterbalanced to reduce this discontinuity. We found that our method also becomes stable as the number of iteration increases. To make our method stable even for a small number of iterations, e.g. one iteration, we devised a smoothed Jacobian:

$$\frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{x}_{ij}} = \begin{cases} k_{ij}\left(1 - \frac{L_{ij}}{|\mathbf{x}_{ij}|}\right) + k_{ij}\frac{L_{ij}}{|\mathbf{x}_{ij}|^3}\mathbf{x}_{ij}\mathbf{x}_{ij}^T, & \text{if } |\mathbf{x}_{ij}| > L_{ij} \\ k_{ij}\frac{L_{ij}}{|\mathbf{x}_{ij}|^2}\mathbf{x}_{ij}\mathbf{x}_{ij}^T f(|\mathbf{x}_{ij}|), & \text{if } \alpha < |\mathbf{x}_{ij}|/L_{ij} < 1 \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

where $f(|\mathbf{x}_{ij}|) = (|\mathbf{x}_{ij}|/L_{ij} - \alpha)/(1-\alpha)$ is a smoothing function and α is a threshold. We set $\alpha = 0.8$ in this paper.

We can consider only strong *type1* springs to compute the $\partial \mathbf{f}_{ij} / \partial \mathbf{x}_{ij}$ in Eq. (8). This causes no problem since *type2* spring is usually weak enough. The resulting method will be almost equally stable with less computation. This is a kind of mixing an explicit scheme for weak springs and an implicit one for strong springs. Similar approach of mixing explicit and implicit scheme is presented in [2]. Our method actually converges to the solution of the implicit method if we iterate it until converges.

A great advantage of our method is that it gives a useful approximation with a fixed small number of Gauss-Seidel iterations per each step; we found that several to ten iterations are enough for wide range of cloth simulation. It is stable regardless of the number of iteration so that we can use large time steps and stiff springs. Therefore, our method is scalable, ranging from one iteration to many iterations until converge. Even with just one iteration, which is almost an explicit method, the stability is an order of magnitude better than that of explicit method. The stability is even further improved as the number of iteration increases.

(Fig. 4)(a), (b), and (c) show snapshots from the simulations by our method with $\eta = 1$, $\eta = 2$, and $\eta = 4$. All of the simulations are stable. We can see the simulation become similar to that of the implicit method as η increases. Although first two images in (Fig. 4)(a) and (b) look different from the last two, notice that they are snapshots from simulation each of which generates a plausible animation. Though simulated motion is different from that of implicit method, simulation with few iterations would be useful in real-time application, since it is very fast like an explicit method.

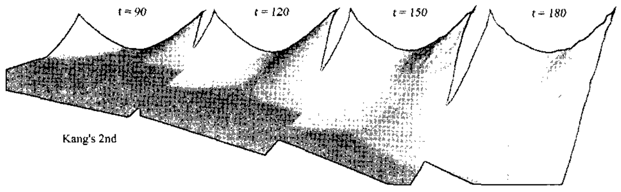
3.1 Artificial inertia and viscosity

We will explain here why our method is stable even with only one iteration, and how we use η as a design parameter. One iteration of our method can be rewritten as

$$\mathbf{v}_i^* = \mathbf{M}^{-1}\left(\mathbf{v}_i + \sum_j \frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{x}_{ij}} \mathbf{v}_j \frac{\Delta t^2}{m}\right) + \mathbf{M}^{-1}\mathbf{F}_i \frac{\Delta t}{m},$$

where $\mathbf{M} = \mathbf{I} + \frac{\Delta t^2}{m}\sum_j \partial \mathbf{f}_{ij}/\partial \mathbf{x}_{ij}$. The first term is the velocity with viscosity: $\mathbf{M}^{-1}((\mathbf{M}-\mathbf{I})\mathbf{v}_i + \sum_j \frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{x}_{ij}}(\mathbf{v}_j - \mathbf{v}_i)\frac{\Delta t^2}{m}$. And, the second term is the acceleration by the forces, which has been diminished by $\mathbf{M}^{-1}$. Compared to the implicit method, the first term contains no more artificial viscosity.

From the second term, the acceleration is scaled by $\mathbf{M}^{-1}$. This is an inertial damping (mass increases) as opposed to the viscous damping in the full-implicit method. We call this damping *artificial inertia*. As we iterate more in our method, this artificial inertia decreases and the artificial viscosity increases. Also, the window size for the viscosity filtering gets larger, eventually the method becomes the implicit method. This artificial inertia properly damps the motion so that we are able to keep the stability without solving a large linear system. More importantly, notice that this artificial inertia as well as the artificial viscosity are limited in the in-plane direction, and do not make a fictitious damping in the out-of-plane direction that resists bending and wrinkling motions. Therefore, our method gives plausible simulation even with a few iteration: This is one of the main contributions of our method.



(Fig. 2) Kang's second method with two Jacobi iterations per step. We set the same condition for the simulation as (Fig. 4). Due to the error of force Jacobian for stretching spring, the simulation becomes unstable beginning from the anchored corner points.

4. Comparisons to other semi-implicit simulation methods

In this section, we briefly study previous methods for cloth simulation that is claimed to be fast and stable. We compare those with our method.

4.1 Desbrun's method

Desbrun et al.[9] approximated the Jacobian of the spring force as $\partial \mathbf{f}_{ij}/\partial \mathbf{x}_{ij} \approx k_{ij}$ by taking only the constant term in Eq. (1). As explained in Section 2, the Jacobian $\partial \mathbf{f}_{ij}/\partial \mathbf{x}_{ij}$ of the spring force is k_{ij} in the direction of the spring. Thus, this approximation is exact for the direction of stretching and compression, but wrong for the direction perpendicular to spring directions. Using this approximation, the Jacobian $\partial \mathbf{F}/\partial \mathbf{x}$ in Eq. (3) is replaced by a constant matrix $\mathbf{H} = (H_{ij})$, where $H_{ii} = -\sum_j k_{ij}$, $H_{ij} = k_{ij}$, which results in an approximated implicit method:

$$(\mathbf{I} - \frac{\Delta t^2}{m}\mathbf{H})\mathbf{v}^* = \mathbf{v} + \mathbf{F}\frac{\Delta t}{m}. \tag{10}$$

The inverse of the system matrix is pre-computed in order to reduce the computation time. However, multiplying the inverted matrix and the system vector still requires a considerable time. This is because the inverse matrix is dense, though the system matrix $\mathbf{I} - \frac{\Delta t^2}{m}\mathbf{H}$ is sparse; in our test with Desbrun's method we used conjugate gradient to solve the sparse matrix, and obtained better results.

A downside of Desbrun's approximation is that a big artificial viscosity arises in the out-of-plane direction, which does not exist in the original implicit method. This will make a damping force that resists bending, wrinkling, overall rotating motion, and even translational motion in the direction of plane normal. (Fig. 4)(i) shows this effect-the velocity of falling cloth is damped a lot compared to the implicit method (in (Fig. 4)(d). They added a post-correction step for compensation by correcting the overall linear and angular momentum. However, the local angular momentum will not be preserved so that fine bending/wrinkling motions can not be formed.

4.2 Kang's first method

Kang et al.[15] introduced an approach similar to our method. They used Desbrun's approximation of force Jacobian and iterate Eq. (8) twice. The resulting approximation derived as follows:

$$\mathbf{v}_i^* = \frac{\mathbf{v}_i + \mathbf{F}_i\frac{\Delta t}{m} + \sum_j k_{ij}\tilde{\mathbf{v}}_j\frac{\Delta t^2}{m}}{1+\sum_j k_{ij}\frac{\Delta t^2}{m}}\text{, where } \tilde{\mathbf{v}}_i = \frac{\mathbf{v}_i + \mathbf{F}_i\frac{\Delta t}{m} + \sum_j k_{ij}\mathbf{v}_j\frac{\Delta t^2}{m}}{1+\sum_j k_{ij}\frac{\Delta t^2}{m}} \quad (11)$$

Notice that the above method does not update $\mathbf{v_i}$ * in place, but uses temporary variable $\widetilde{\mathbf{v_i}}$. This is an important difference from our methods. We already mentioned that in-place updating of $\mathbf{v_i}$ * is crucial for achieving the stability. Fortunately, the above method is stable due to the scalar-valued approximation k_{ij} of the force Jacobian. We think that the in-place updating will enhance the quality of simulation also in this case.

Even with this simple formulation the method is unconditionally stable for any value of time step and stiffness. Regrettably, we found the stability is not for free: The method incurs a severe damping as spring stiffness (k_{ij}) grows.

4.3 Kang's second method

Kang et el.[13] noticed that the scalar approximation of force Jacobian causes a severe damping. They also used a vector-valued force Jacobian in their second method so that the unnecessary damping in the out-of-plane direction could be reduced. Our method is an improvement to Kang's second method by using Gauss-Seidel iteration and a smoothed force Jacobian. Here, we would like to state the differences between two methods.

We employed 3×3 block Gauss-Seidel iteration for handling Eq. (8), while 3×3 block Jacobi iteration is used in [13]. Both of Guass-Seidel and Jacobi iterations eventually converges to the same solution and make the cloth simulation stable when we do have a enough number of iterations. However, here we have room for limiting the number of iterations from one to ten for a real-time simulation, and this makes a difference between Gauss-Seidel and Jacobi iterations.

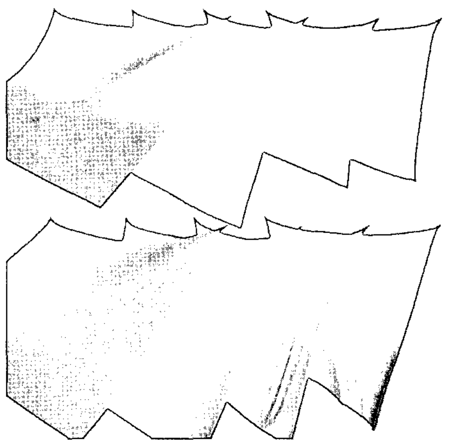
We found that the cloth simulation with Gauss-Seidel is more stable than with Jacobi iteration when the number of iteration is limited. Especially, with an odd number of Jacobi iterations, the stability of Kang's second method is quite poor. (Fig. 4)(e) and (f) show this effects. Kang's second method (with odd η) becomes unstable for stiff springs or large Δt. However, our method is stable for any η and even with only one iterations. We recommend to use an even number of Jacobi iteration for Kang's second method. Also with an even number of (Gauss-Seidel or Jacobi) iterations, our method is slightly more stable and converges little faster than Kang's second method (Compare (Fig. 4)(c) and (g).

Devising a smoothed Jacobian (Eq. (9) is another improvement of our method over Kang's second method. Both of ours and Kang's are unstable with Choi's asymmetric spring force function (Eq. (6) and its Jacobian (Eq. (7), which is necessary for improving the stability and the plausibility of cloth animation [6]. We were able to employ the asymmetric force function (Eq. (6) successfully by devising a smoothed force Jacobian (Eq. (9).

The following approximate force Jacobian is used in Kang's second method for simpler computation by assuming small elongation/compression of springs.

$$\frac{\partial \mathbf{f}_{ij}}{\partial \mathbf{x}_{ij}} = k_{ij}\frac{\mathbf{x}_{ij}\mathbf{x}_{ij}^T}{|\mathbf{x}_{ij}|^2} \quad (12)$$

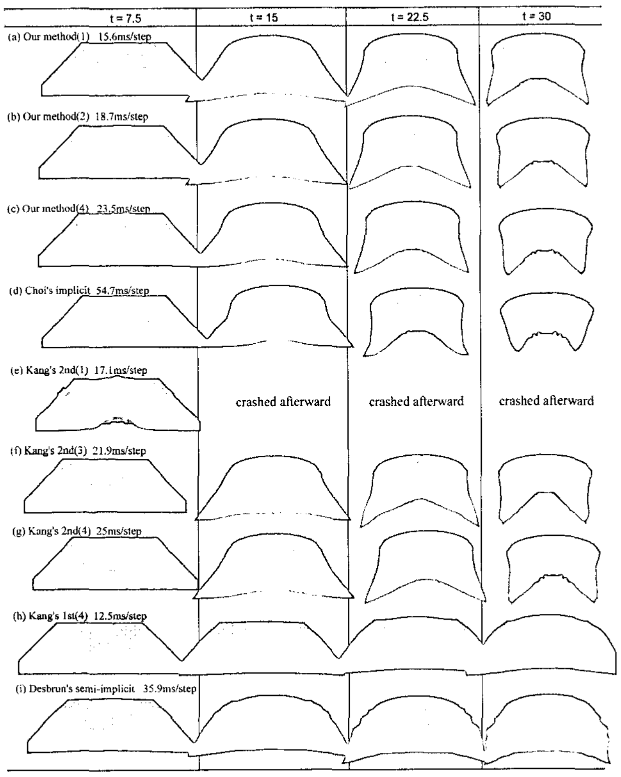
We also tested the idea of combining the asymmetric spring (Eq. (6) and the above approximate Jacobian. We found this combination also gives a good cloth simulation but with significant drawbacks. It makes unnecessary damping for compression of springs, which may result in removing of fine wrinkles. On the contrary, it makes less damping for the elongation compared to that of the original force Jacobian. This damping is crucial for stabilizing the cloth simulation. Kang's method with the approximate Jacobian (Eq. (12) can be unstable when the spring is very stretched. (Fig. 2) is an example of showing this effect, where only two corner points of a cloth are anchored. The simulation begins to break around the stretched region of the anchored corners.



(Fig. 3) Draping test with Choi's implicit (top) and our method (bottom). Only two corner points are anchored. Large stiffness of 1000 is used for *type1* springs to prevent elongations around the anchor point. Our near-explicit method is iterated six times for every simulation step. The computing times of the implicit and our methods are 200msec/step and 45msec/step in Pentium 4 2.4GHz, respectively.

5. Experimental results

(Fig. 4) shows cloth simulation results of our near-explicit method and other various methods from [6, 9, 15, 13]. The cloth model consists of 80 × 80 points of unit mass. We set the *type1* spring stiffness $k_{ij} = 50$, the *type2* spring stiffness $k_{ij} = 0.1$, the time step $\Delta t = 0.25$, the gravity $g = 0.03$, and the damping factor $d = 0.05$. Though Kang's first method has the fastest computing time per step, actually the simulation is very slow due to the big artificial inertia. We claim that our method is practically fastest semi-implicit method. Also the simulation quality is comparable to that of the implicit method; Our method has slightly more artificial damping than the implicit method.


t = 7.5
t = 15
t = 22.5
t = 30
(a) Our method(1) 15.6ms/step
(b) Our method(2) 18.7ms/step
(c) Our method(4) 23.5ms/step
(d) Choi's implicit 54.7ms/step
(e) Kang's 2nd(1) 17.1ms/step
crashed afterward
crashed afterward
crashed afterward
(f) Kang's 2nd(3) 21.9ms/step
(g) Kang's 2nd(4) 25ms/step
(h) Kang's 1st(4) 12.5ms/step
(i) Desbrun's semi-implicit 35.9ms/step

(Fig. 4) Cloth simulations results of various methods. The size of mass-spring model is 80 × 80 and each node has a unit mass. We set the same conditions for each method: *type1* spring stiffness $k_{ij} = 50$, *type2* spring stiffness $k_{ij} = 0.1$, $\Delta t = 0.25$, gravity = 0.03, damping = 0.05. The computation time per step is measured in Pentium 4 2.4GHz. (a)-(c) Our methods, (d) Choi's implicit method, (e)-(g) Kang's second methods, (h) Kang's first method, and (i) Desbrun's semi-implicit method. The parenthesized number after the method name indicates the number of Gauss-Seidel or Jacobi iteration per each simulation step. Our method is stable like an implicit method, and also faster than other method while resulting a plausible animation.

However, it has much less damping than other semi-implicit methods.

A draping test is provided in (Fig. 3). Only two corner points are anchored, and we set spring stiffness very high (1000) to prevent elongation around the anchored points. Since our method is stable for very large stiffness, we do not need any post-step of limiting spring elongation, which was the case in [17]. We compared our method and Choi's[6] implicit method. Our method worked similar to the implicit method except the velocity is slightly slow in our method due to artificial inertia. However, our method is equally stable and four times as fast as the implicit method in this example.

6. Conclusion

We presented a scalable semi-implicit method for real-time cloth simulation, which is an improvement of Kang's method[13]. The proposed method is explicitly represented in simple closed form, and thus very fast and defeats any implicit methods in terms of speed. Our method is stable and thus able to use large time steps and stiff springs despite its simplicity. This stability is achieved by a proper amount of artificial viscosity and inertia, which acts only in the in-plane direction where tension is high. Thus the method minimally resists bending and wrinkling motions of cloth, and produces a less damped motion.

We compared our method with the previous semi-implicit methods; they have an extra damping, unnecessary for stabilization, not found in the implicit method. Although our method has more damping (due to the artificial inertia) than the implicit method, it is somehow minimal and far less than that of the previous semi-implicit methods. Also, we found our method is more stable than the previous semi-implicit methods.

References

[1] David Baraff and Andrew Witkin. Large steps in cloth simulation. In Proceedings of SIGGRAPH '98, Computer Graphics Proceedings, Annual Conference Series, pp.43-54. ACM, ACM Press/ACM SIGGRAPH, 1998.

[2] E. Boxerman and U. Ascher. Decomposing cloth. In Procedings of Eurographics/SIGGRAPH Symposium on Computer Animation '04, 2004.

[3] D.E. Breen, D.H. House, and M.J. Wozny. Predicting drape of woven cloth using interating particle. In Proceedings of SIGGRAPH '94, Computer Graphics Proceedings, Annual Conference Series, pages 365-372. ACM, ACM Press / ACM

SIGGRAPH, 1994.

[4] R. Bridson, S. Marino, and R. Fedkiw. Simulation of clothing with folds and wrinkles. In Procedings of Eurographics/ SIGGRAPH Symposium on Computer Animation '03, 2003.

[5] M. Carignan, P. Volino, and N. Magnenat-Thalmann. Dressing animated synthethic actors with complex deformable cloth. In Proceedings of SIGGRAPH '92, Computer Graphics Proceedings, Annual Conference Series, pp.99-104. ACM, ACM Press/ACM SIGGRAPH, 1992.

[6] Kwang-Jin Choi and Hyeong-Seok Ko. Stable but responsive cloth. In Proceedings of SIGGRAPH '02, Computer Graphics Proceedings, Annual Conference Series, pp.604-611. ACM, ACM Press/ACM SIGGRAPH, 2002.

[7] F. Cordier and N. Magnenat-Thalmann. Real-time animation of dressed virtual humans. Computer Graphics Forum (Eurographics 2002), Vol.21, No.3, pp.327-336, September, 2002.

[8] F. Cordier and N. Magnenat-Thalmann. A data-driven approach for real-time clothes simulation. In Proceedings of 12th Pacific Conference on Computer Graphics and Applications (PG 2004), October, 2004.

[9] Mathieu Desbrun, Peter Schr¨oder, and Alan Barr. Interactrive animation of structured deformable objects. In Proceedings of Graphics Interface '99, pp.1-8, 1999.

[10] R. Gershbein, L. Cutler, X.C. Wang, C. Curtis, E. Maigret, L. Prasso, and P. Farson. An art-directed wrinkle system for CG character clothing. In Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2005). ACM Press, 2005.

[11] M. Hauth, O. Etzmuß, and W. Straßer. Analysis of numerical methods for the simulation of deformable models. The Visual Computer, Vol.19, No.7-8, pp.581-600, December, 2003.

[12] Y.-M. Kang and H.-G. Cho. Bilayered approximate integration for rapid and plausible animation of virtual cloth with realistic wrinkles. In Proceedings of Computer Animation, pages 203-214. IEEE Computer Society, 2002.

[13] Young-Min Kang and Hwan-Gue Cho. Complex deformable objects in virtual reality. In Proceedings of VRST 2002, pp.49-56, 2002.

[14] Young-Min Kang and Hwan-Gue Cho. Real-time animation of complex virtual cloth with physical plausibility and numerical stability. Presence, Vol.13, No.6, pp.668-680, 2004.

[15] Young-Min Kang, Jeong-Hyeon Choi, Hwan-Gue Cho, and Chan-Jong Park. Fast and stable animation of cloth with an approximated implicit method. In Proceedings of Computer Graphics International 2000, pp.247-255, 2000.

[16] M. Desbrun M. Meyer, G. Debunne and A. Barr. Interactive animation of cloth-like objects in virtual reality. Journal of Visualization and Computer Animation, Vol.12, No.1, pp.1-12, 2001.

[17] Xavier Provot. Deformation constrraints in a mass-spring model to describe rigid cloth behavior. In Proceedings of Graphics Interface '95, pp.147-154, 1995.

[18] D. Terzopoulos and K. Fleischer. Modeling inelastic deformation: Viscoelasticity, plasticity, fracture. In Proceedings of SIGGRAPH '88, Computer Graphics Proceedings, Annual Conference Series, pp.269-278. ACM, ACM Press / ACM SIGGRAPH, 1988.

[19] P. Volino and N. Thalmann. Comparing efficiency of integration methods for cloth simulation. In Proceedings of Computer Graphics International 2001 (CGI 2001), pp.265-274, 2001.

김 명 준

e-mail : mjkim@ewha.ac.kr

1989년 한국과학기술대학 전산학과(학사)

1991년 한국과학기술원 전산학과(석사)

1996년 한국과학기술원 전산학과(박사)

1996년~1997년 University of Washington 연구원

1997년~1998년 시스템공학연구소 선임연구원

1998년~2000년 한국전자통신연구원 선임연구원

2000년~2001년 ㈜ 엔룩스 연구소장

2001년~현재 이화여자대학교 디지털미디어학부 조교수

관심분야 : 컴퓨터그래픽스, 물리기반 애니메이션, 게임 프로그래밍