# A New Implicit Integration Method for Low Damped Cloth Simulation

Seungwoo Oh[1], Junghyun Ahn and Kwangyun Wohn
**Virtual Reality Laboratory, EECS, Korea Advanced Institute of Science and Technology**

**Abstract**

*This paper discusses a new technique that reproduces a stable cloth without introducing excessive damping artifacts. Semi-implicit integration methods have been widely used in cloth simulations for their high stability and speed. Artificial internal damping forces are derived from the linearization process of semi-implicit integration. This makes the system highly stable. However, the artificial internal damping forces are generated with respect to not only internal deformations but also rotational rigid motions. This damping artifact significantly degrades the realism of the cloth simulation. Hence, we propose a new implicit integration method to decrease this artifact. The artificial internal damping forces are computed by considering the pure internal deformations alone, and are stably incorporated into the dynamical system. As a result, our simulator could reproduce a wide range of cloth materials without the excessive damping artifacts in real-time.*

Keywords: Computer Animation, Cloth Simulation, Implicit Integration

## 1. Introduction

This work was intended for a development of garment-related e-commerce system [6][23]. In this system, a variety of cloth materials need to be simulated in real-time for providing realism to customers.

Since the pioneering work of Baraff and Wikin [1], semi-implicit integration has been widely used for making clothing animations. This is due to better stability and speed of the implicit scheme over explicit one. Several integration methods have been analyzed in terms of their speed and accuracy in [21], [12] and [13].

The semi-implicit integration method[1][5][20][8][2][4] linearizes the non-linear dynamics equations. From this linearization, an internal damping force is introduced [7]. We will refer to this as an artificial internal damping (AID) force.

Several artifacts arise when the AID forces are generated with respect to the rotational motions without the internal deformations. Such artifacts have been reported in several literature [21][8]: wrinkles did not form or disappear, and cloths did not fall according to their weights, and movements even stopped. In real-time simulations, a limited number of iterations for solving the linear system have to be performed to reduce the computational time. This also becomes a source of the excessive damping artifacts.

Several integration schemes can alleviate these artifacts: Newmark method [11], semi-implicit integration with second-order backward difference formula [20][5], and a generalized-α method [18]. Nevertheless, the excessive damping problems remain unsolved, since all these methods undergo the linearization process.

The use of Newton's method may eliminate these problems, because this method can solve the non-linear system (see [12] for Newton's method adapted in cloth simulation). This method, however, has to repeat the solving procedure numerous times. Compared with this, the semi-implicit scheme requires solving the linear system once. Thus, the semi-implicit scheme rather than Newton's method is more suitable for real-time cloth simulations. For this reason, we decided to adopt the semi-implicit scheme.

In this paper, we propose a new linearization scheme of semi-implicit integration that can avoid the damping artifacts. From this linearization, a new AID force is introduced. This force is computed by considering solely the internal deformation so that this force is not generated with respect to the rigid body motion. To put it more concretely, since we use spring-mass model, this damping force is evaluated based on the change in spring length along the current velocity path. We refer to this as a pure artificial internal damping (PAID) force.

Experimental results demonstrated that the proposed method is very stable and can be used for reproducing a wide range of cloth materials in real-time.

Before we discuss our main concern, it will be useful to review the spring-mass model (Section 2) and the semi-

[1] Address: 373-1 Kusong-Dong, Yusong-Gu, Taejon 305-701, VR Lab., EECS, KAIST, KOREA, Phone: +82-42-869-3572, Fax: +82-42-869-8830, E-mail: redmong@vr.kaist.ac.kr

implicit integration (Section 3). In order to introduce the AID and the PAID force (Section 4), we have to clarify understanding of Jacobian matrix first (Section 3). How to incorporate the PAID force into the dynamical system is discussed in Section 5. Finally, we show and experimental results and conclude our work.

## 2. Cloth Modeling

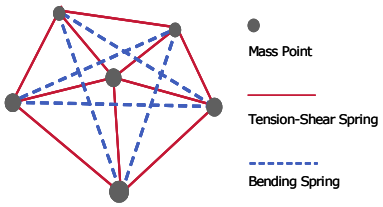



**Figure 1:** *Mass-spring model*

The present work employs a mass-spring model, shown in Figure 1, which is widely used for cloth simulations. Tension-shear springs are connected between two vertices on each edge of the triangular mesh consisting of cloth. Bending springs are connected between the two vertices across each edge. This model has been adopted in [19], [5], [4], [7] and [15], since it is easy to implement, and yields visually plausible results.

Previous works have modeled the spring force in various ways. The simplest way is to generate the repulsive force linearly with respect to the change in length. This linear spring model has been used by [19], [4], [7] and [15]. As the linear spring force is estimated well by implicit integration, these simulations yield very stable results. Real cloth materials, however, have non-linear properties.

Non-linear spring models are used for the accurate analysis of cloth behavior, such as buckling, folding, and stretching. Breen et al. [3] demonstrated the realistic static draping of cloth using an approximated polynomial model based on experimental measurements. Eberhardt et al. [10] extended this work to a dynamic simulation. Choi and Ko [5] derived a non-linear model based on an assumption that the curvature and length of the springs are maintained when they decrease below their rest length. By using this force, they achieved a stable and realistic simulation of wrinkles. We also use the similar non-linear (but, piecewise-linearly approximated) model.

## 3. Semi-Implicit Integration

Once the position, $\mathbf{x}$, velocity, $\mathbf{v}$, and the force, $\mathbf{f}$, of each mass-point are known at the current time, $t^{(n)}$, then we need to apply an integration method to obtain the states in the following time, $t^{(n+1)}$. Implicit Euler integration forms the differential (non-linear) equations as follows,

$$\begin{aligned} \Delta\mathbf{x} &= h\mathbf{v}^{(n+1)} \quad , \ \Delta\mathbf{x} = \mathbf{x}^{(n+1)} - \mathbf{x}^{(n)} \\ \Delta\mathbf{v} &= h\mathrm{M}^{-1}\mathbf{f}^{(n+1)} \quad , \ \Delta\mathbf{v} = \mathbf{v}^{(n+1)} - \mathbf{v}^{(n)} \end{aligned} \tag{1}$$

where $h$ is the simulation time step. Each vector is $3N$-dimensional when the cloth is composed of $N$ mass-points. A value of the mass of each particle is included in a diagonal matrix, $\mathbf{M}$.

The semi-implicit scheme linearizes above equations through the following first-order approximation:

$$\mathbf{f}^{(n+1)} = \mathbf{f}^{(n)} + \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^{(n)} \Delta\mathbf{x} + \left(\frac{\partial \mathbf{f}}{\partial \mathbf{v}}\right)^{(n)} \Delta\mathbf{v} \ . \tag{2}$$

Substituting Equation (2) into Equation (1), we, finally, obtain a linear system as follows

$$\left(\mathbf{M} - h\left(\frac{\partial \mathbf{f}}{\partial \mathbf{v}}\right)^{(n)} - h^2\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^{(n)}\right)\Delta\mathbf{v} = h\left(\mathbf{f}^{(n)} + h\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^{(n)}\mathbf{v}^{(n)}\right) \tag{3}$$

To solve this linear system, we evaluate all the forces and the corresponding Jacobian matrices, and then repeat Conjugate Gradient (CG) iterations (see [1], [5], [20] and [4] for more detail).

### Jacobian Matrix

The Jacobian matrix plays an important role in reproducing the realistic behaviors of cloth. The material properties, wrinkle formation, and the global behavior of a cloth are closely dependant on the Jacobian matrix. In this section, we discuss on how the Jacobian matrix encapsulates the behavioral properties of a cloth.

In the process to solve the linear system, i.e., the process that determines the deformations of the cloth in the following time, the Jacobian matrix plays two roles. The first is to determine the behavior between two mass-points, and the second is to determine the behavior of the mass-point itself. We refer to these as an inter-Jacobian matrix, and as a self-Jacobian matrix, respectively.

$$\mathrm{J}_{ij} = \frac{\partial \mathrm{f}_i}{\partial \mathrm{x}_j}$$

- The inter-Jacobian matrix: This equation represents the movement force at a mass, $j$, transferring to a mass-point, $i$. A large magnitude of the Jacobian results in a strong force, since the force exerted on the mass-point, $i$, is evaluated multiplying the Jacobian by the change in position. In terms of the direction, an anisotropic Jacobian yields forces with different magnitude for each basis directions.

$$\mathrm{J}_{ii} = \frac{\partial \mathrm{f}_i}{\partial \mathrm{x}_i}$$

- The self-Jacobian matrix: This equation represents the movement force that a mass-point, $i$, exerts on itself. In the process of solving a linear system, the inverse of this matrix is used, and the inverse matrix represents how far the mass-point moves according to the change in the force. The movement is dependent on the magnitude and direction of the Jacobian.

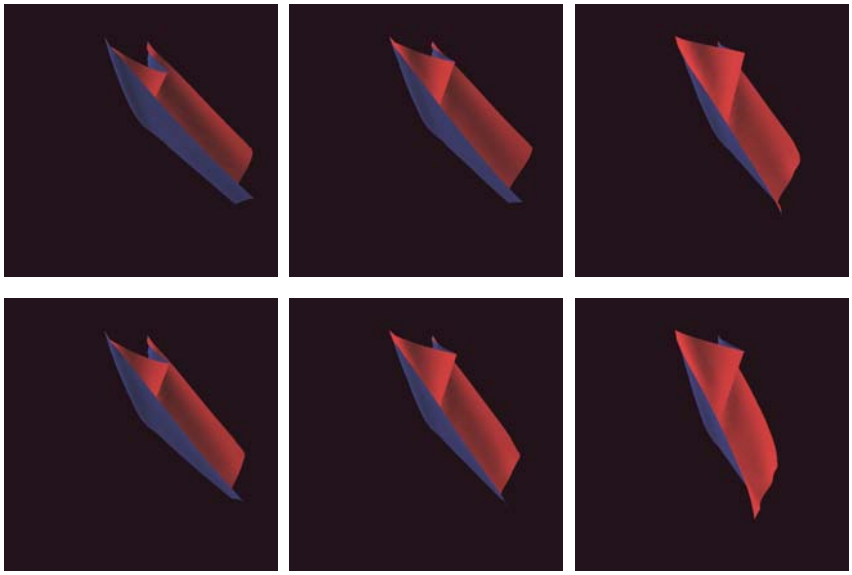


**Figure 2:** *Snapshots from the simulation using our method (top), and the simulation using the implicit Euler method (bottom). From left to right, stiffer materials were used. In these simulations, the wind blew from beginning and stopped after 1 sec. All the snapshots are captured at 1 sec after beginning. Our method preserved more rotational momentum.*

We would like to explain how the two above Jacobian matrices affect the behavior of cloth by giving the example of Jacobi iteration, which is one of the methods used to solve a linear system (This is also valid in the CG method). Jacobi iteration method repeats the process where the next state of a mass-point is updated according to the current status of its nearest neighbors. The process for a mass-point, $i$, is to solve the following equation [15]

$$\left(\mathbf{M}_i - h^2\left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_i}\right)^{(n)}\right)\Delta \mathbf{v}_i = \left(h\left(\mathbf{f}_i^{(n)} + h\left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}}\right)^{(n)} \mathbf{v}^{(n)}\right) + h^2 \sum_{i \neq j}\left(\left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_j}\right)^{(n)} \Delta \mathbf{v}_j\right)\right)$$

where the right-hand portion represent a multiplication of the net force by the simulation time step. The net force includes the force generated when neighboring mass-points move. This force is computed according to the inter-Jacobian matrices between a mass-point, $i$, and its neighbors. In this way, neighboring mass-points exert influence on the movement of a mass-point.

Finally, the change in velocity of a mass-point, $i$, is obtained by multiplying the right-hand terms, $\mathbf{r}_i$, of the above equation by the inverse of the matrix, including the self-Jacobian, as follows

$$\Delta \mathbf{v}_i = \left(\mathbf{M}_i - h^2\left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_i}\right)^{(n)}\right)^{-1} \mathbf{r}_i .$$

From the above equation, we can see that the movement of a mass-point is determined by its self-Jacobian. If the Jacobian matrix includes any derivatives of the unnecessary forces, the mass point will slow down along the directions of the derivatives.

We can easily see that the Jacobian matrix can not render non-linear interaction between mass-points for its linearity. The rotational interactions are neglected and this lead to damped movements of mass-points in rigid body motions. This fact becomes clear by considering the Jacobian matrix for a spring tension.

The Jacobian of the spring tension with respect to the change in position, is given by

$$\mathbf{J}_{ij}^{spring} = \frac{\partial \mathbf{f}_i^{spring}}{\partial \mathbf{x}_j} = k_{ij}\left[(1 - \frac{l_{ij}^0}{|\mathbf{x}_{ij}|})(\mathbf{I}_3 - \frac{\mathbf{x}_{ij}\mathbf{x}_{ij}^T}{|\mathbf{x}_{ij}|^2}) + \frac{\mathbf{x}_{ij}\mathbf{x}_{ij}^T}{|\mathbf{x}_{ij}|^2}\right] \quad (4)$$

where $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ [5]. Note that the Jacobian is defined as the summation of a projection matrix along the spring direction and a projection matrix along a plane orthogonal to the spring direction. Thus mass-points interact linearly along these basis directions.

From what has been discussed above, we can draw that using [7][8] or including [20] isotropic matrix as Jacobian like in several previous works worsens the damping problems. This is because the use of isotropic matrix makes mass-points interact equivalently along all directions, regardless of the directions of the internal forces.

## 4. Damping Artifacts from Linearization

We can classify the sources of the damping artifacts into two: the damping force by the current velocity (AID force) and the damping force by the change in velocity during a simulation time-step.

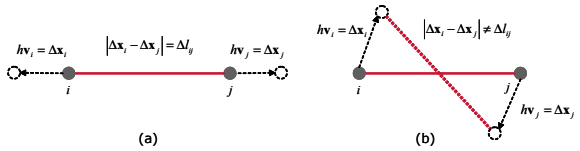



**Figure 3:** *1D spring (a). The change in magnitude of the position is the same as the length of spring. Rotating spring in 2D (b). The change in magnitude of the position is different from the length of spring.*

We first introduce the AID force. The right-hand term of Equation (3) includes the AID force [7]. The following equation represents the AID force for the mass-point, $i$ :

$$\left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}}\right)^{(n)} h\mathbf{v} = \sum_{i \neq j}\left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_j}\right)^{(n)} h\left(\mathbf{v}_j - \mathbf{v}_i\right)$$
$$\mathbf{f}_{ij}^{AID} = \left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}_j}\right)^{(n)} h\left(\mathbf{v}_j - \mathbf{v}_i\right) \quad . \quad (5)$$

We note that an internal damping force between any two mass-points is computed by multiplying the inter-Jacobian matrix by the difference between the two changes in the positions during a simulation time-step. Therefore we can say that this force has the same order of magnitude as the spring tension. This fact makes the AID force distinguished from the material intrinsic internal damping force. Thus, the AID force rather than the intrinsic one can cause the severe damping artifacts.

In the case of a 1D spring, the corresponding AID force make the length of the spring fixed regardless of the current velocities (see Figure 3). Thus the spring must always be stable. This is because this AID force has the same magnitude as the spring tension with respect to the change in length along the current velocity path. We define the pure artificial internal damping (PAID) force as this AID force.

In higher dimension, an AID force, however, can cause the excessive damping problems. See Figure 3-(b). The spring is rotating with no internal deformations. No internal forces must be generated here. Nevertheless, a damping force is generated because the self-Jacobian matrix is not null and the change in spring length is not matched with the difference between the two changes in the positions.

The damping force by the change in the velocity during a simulation time-step also has the similar form as the AID force. The left-hand term of Equation (3) includes

$$\left(\frac{\partial \mathbf{f}_i}{\partial \mathbf{x}}\right)^{(n)} h\Delta\mathbf{v} \, .$$

This is the same as changing $\mathbf{v}$ into $\Delta\mathbf{v}$ in Equation (5). The damping artifacts due to this force appear during the process of solving the linear system.

All previous works have used these damping forces introduced from the linearization as they are. Although we can not avoid the damping artifacts due to the change in velocity, we may avoid the damping artifacts due to AID force. This is because the AID force is given explicitly in the semi-implicit formula while the force by the change in velocity is unknown. If the PAID force is used instead of this AID force, then we can eliminate one of the two sources of the artifacts. Thus we decided to adopt the PAID force for our simulator.

## 5. Semi-implicit Integration with PAID Force

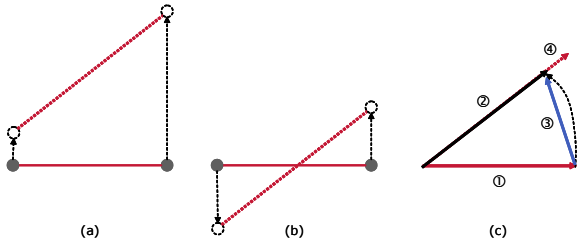



**Figure 4:** *The motion of a spring, including the rigid body motion and the internal deformation (a). The motion of a spring in its nontranslating reference frame (b). The change in force under the motion (c). Key: ① = the force at the current time; ②+④ = the force at the following time; ② = the force changed solely by the rotation at the following time; ③ = the change in force under the rotation during the simulation time step, and ④ = the change in force by pure internal deformation( PAID force).*

### Computation of PAID Force

The PAID force is computed as the spring tension with respect to the change in length along the current velocity path (④ of Figure 4-(c)).

$$\mathbf{l}_{ij} = (|\,\hat{\mathbf{x}}_{ij}\,| - |\,\mathbf{x}_{ij}\,|)\frac{\hat{\mathbf{x}}_{ij}}{|\,\hat{\mathbf{x}}_{ij}\,|}$$
$$\mathbf{f}_{ij}^{PAID} = -k_{ij}\mathbf{l}_{ij}$$

where $\hat{\mathbf{x}}_{ij} = (\mathbf{x}_i + h\mathbf{v}_i) - (\mathbf{x}_j + h\mathbf{v}_j)$ and $\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$ . This idea, based on the local reference frame, has been applied not only in mechanical engineering field but also in several computer graphics related works [16][9]. While these related works were concerned with the exact computation of stress and strain, we deal with the exact and stable computation of the AID force.

### Linearization

We would like to introduce our linearization scheme for the semi-implicit integration with the PAID force. We first note that all the force terms in this section mean only the internal forces for the ease of explaining. Equation (2) shows the traditional linearization scheme. However, what we really want is as follows

**Figure 5**: *Garments simulated by our method (top) and the implicit Euler method (bottom).*

$$\mathbf{f}^{(n+1)} = \mathbf{f}^{(n)} + \Delta\mathbf{f}_{\mathrm{x}} + \Delta\mathbf{f}_{\mathrm{v}}$$

where $\Delta\mathbf{f}_{\mathrm{x}} = \Delta\mathbf{f}_x(\mathbf{x}^{(n)}, \Delta\mathbf{x})$ and $\Delta\mathbf{f}_v = \Delta\mathbf{f}_x(\mathbf{v}^{(n)}, \Delta\mathbf{v})$ mean the change in force with respect to the change in position and velocity, respectively. Here, we are concerned with only the position part which is related to the PAID force. By the linear approximation, $\Delta\mathbf{x} = (\mathbf{v}^{(n)} + \Delta\mathbf{v})h$, we derive $\Delta\mathbf{f}_{\mathrm{x}} = \Delta\mathbf{f}_x(\mathbf{x}^{(n)}, (\mathbf{v}^{(n)} + \Delta\mathbf{v})h)$, and then split this into the change in force with respect to $h\mathbf{v}^{(n)}$ and $h\Delta\mathbf{v}$ as follows

$$\Delta\mathbf{f}_{\mathrm{x}} = \Delta\mathbf{f}_x(\mathbf{x}^{(n)}, h\mathbf{v}^{(n)}) + \Delta\mathbf{f}_x(\mathbf{x}^{(n)} + h\mathbf{v}^{(n)}, h\Delta\mathbf{v})$$

where the PAID force corresponds to $\Delta\mathbf{f}_x(\mathbf{x}^{(n)}, h\mathbf{v}^{(n)})$. The remaining term, $\Delta\mathbf{f}_x(\mathbf{x}^{(n)} + h\mathbf{v}^{(n)}, h\Delta\mathbf{v})$, is unknown. This term can be linearly approximated as

$$\Delta\mathbf{f}_x(\mathbf{x}^{(n)} + h\mathbf{v}^{(n)}, h\Delta\mathbf{v}) \approx \left(\frac{\partial\mathbf{f}_x}{\partial\mathbf{x}}\right)^{(n)} h\Delta\mathbf{v}\,.$$

Now, only the Jacobian matrix remains unsolved. Using the difference quotient, we can obtain the following equations.

$$\left(\frac{\partial\mathbf{f}_x}{\partial\mathbf{x}}\right)^{(n)} \approx \frac{\Delta\mathbf{f}_x}{\Delta\mathbf{x}} = \frac{\Delta\mathbf{f}_x(\mathbf{x}^{(n)}, h\mathbf{v}^{(n)})}{\Delta\mathbf{x}} + \frac{\Delta\mathbf{f}_x(\mathbf{x}^{(n)} + h\mathbf{v}^{(n)}, h\Delta\mathbf{v})}{\Delta\mathbf{x}}$$

Based on the direction of springs at $\mathbf{x}^{(n)}$ (① of Figure 4-(c)) and $\mathbf{x}^{(n)} + h\mathbf{v}^{(n)}$ (② of Figure 4-(c)), we can approximate each term as the Jacobian matrix of the spring (see Equation (4)).

$$\frac{\Delta\mathbf{f}_x(\mathbf{x}^{(n)}, h\mathbf{v}^{(n)})}{\Delta\mathbf{x}} \approx \mathbf{J}^{spring}(\mathbf{x}^{(n)})$$

$$\mathbf{J}_{ij}^{spring}(\mathbf{x}^{(n)}) = \frac{k_{ij}}{2}\left[(1 - \frac{l_{ij}^0}{|\,\mathbf{x}_{ij}\,|})(\mathbf{I}_3 - \frac{\mathbf{x}_{ij}\mathbf{x}_{ij}^T}{|\,\mathbf{x}_{ij}\,|^2}) + \frac{\mathbf{x}_{ij}\mathbf{x}_{ij}^T}{|\,\mathbf{x}_{ij}\,|^2}\right]$$

$$\frac{\Delta\mathbf{f}_x(\mathbf{x}^{(n)} + h\mathbf{v}^{(n)}, h\Delta\mathbf{v})}{\Delta\mathbf{x}} \approx \mathbf{J}^{spring}(\hat{\mathbf{x}}^{(n)})$$

$$\mathbf{J}_{ij}^{spring}(\hat{\mathbf{x}}) = \frac{k_{ij}}{2}\left[(1 - \frac{l_{ij}^0}{|\,\hat{\mathbf{x}}_{ij}\,|})(\mathbf{I}_3 - \frac{\hat{\mathbf{x}}_{ij}\hat{\mathbf{x}}_{ij}^T}{|\,\hat{\mathbf{x}}_{ij}\,|^2}) + \frac{\hat{\mathbf{x}}_{ij}\hat{\mathbf{x}}_{ij}^T}{|\,\hat{\mathbf{x}}_{ij}\,|^2}\right]$$

where we multiply the stiffness by 0.5 for the scale normalization. Note that the above two Jacobian matrices are aligned to the directions of the spring tension and the PAID force, respectively. In other words, we already obtained the Jacobian matrices that can linearly predict the changes in the two largest forces. For this reason, the system is stable despite adding of the PAID force.

Finally, by computing the material intrinsic damping force with respect to $\Delta\mathbf{v}$, $\Delta\mathbf{f}_{\mathrm{v}}$, as the isotropic force, we obtain a linear system as follows

$$\left(\mathbf{M} - hk^{damp}\mathbf{I} - h^2(\mathbf{J}^{spring}(\mathbf{x}^{(n)}) + \mathbf{J}^{spring}(\hat{\mathbf{x}}^{(n)}))\right)\Delta\mathbf{v} = h\left(\mathbf{f} + \mathbf{f}^{PAID}\right)$$

Note, if $\mathbf{x}^{(n)}$ is aligned with $\hat{\mathbf{x}}^{(n)}$, namely $\mathbf{x}^{(n)}$ is aligned with $\mathbf{v}^{(n)}$, the system above is identical with the linear system of implicit Euler integration (Equation (3)).

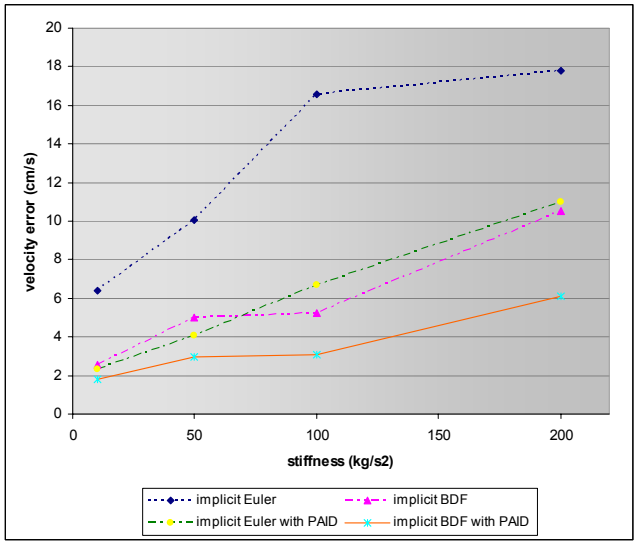
## 6. Implementation and Experimental Results

We have implemented our cloth simulator using both of Java and C++. The simulator and the result movies can be found on our web site [23].

For the dynamic collision detection, we have applied Oh's linear simplification method [17] to triangle-vertex collision detection. We detect collisions between the moving triangles on the human body and the moving particles on the cloth. To prune the large number of collision-possible triangles, a voxel structure has been used. For the collision response, we have adopted Baraff's method [1].

To demonstrate the advantages of our approach, we analyzed the accuracies of the implicit Euler integration, the second BDF implicit integration, the implicit Euler with the PAID force, and the second BDF implicit integration with the PAID force. First, in order to compare the damping effects with respect to the rigid body motions, we performed the experiment where a rectangle cloth flapped by wind as shown in Figure 2. Table 1 shows the average of the velocity errors between data of each method and original data. Original data was obtained by using explicit Euler method with very small time step (0.00001s). The velocity error has been adopted as the error measure, because the position errors are accumulated while an animation runs. We also compared the damping effects in the formation and disappearance of wrinkles through animations of clothed avatar (Figure 5). The use of the PAID force resulted in more natural animations of wrinkles. Figure 6 shows each image captured from dance and ballet animations produced by our method. In these animations, the garment was composed of roughly 8000 springs and 1400 mass-points. We were able to simulate this in real-time (about 30 fps). All experiments were performed on a PC with a 2.4GHz Pentium IV processor.

**Table 1 . Error comparison**




## 7. Conclusion

In the present paper, we have proposed a new semi-implicit integration scheme that can decrease the excessive damping artifacts in rotational rigid motions. We discovered the causes of the artifacts: the damping forces by the current velocity (the AID force) and by the change in the velocity during a simulation time-step. Then we suggested a solution for the artifacts due to the AID force. A new artificial internal damping force (the PAID force) was introduced. This force is not generated with respect to the rigid body motions, differently from the previous works. Consequently, our simulator can reproduce a wide range of cloth materials, including very stiff one, without any severe damping artifacts in real-time.

This work has discussed on an approach to solve a rotation-related problem among many non-linear problems by using a linear solver. However, the non-linear problems such as fast and stable animations of complex wrinkles still remain unsolved. In future, we will study on a linear approach for such problems. Moreover, we would like to explore for a level of detail approach for a really real-time cloth simulation.

## Acknowledgements

This work has been funded by Korea Advanced Institute of Science and Technology.

## REFERENCES

[1] Baraff D. and Witkin A.: Large steps in cloth simulation, Computer Graphics (Proc. SIGGRAPH), pp. 43-54, 1998.

[2] Bridson R., Fedkiw R.P., Anderson J.: Robust treatment of collisions, contact, and friction for cloth animation. Proc. SIGGRAPH, pp. 594-603, 2002.

[3] Breen, D.E., House, D.H., and Wozny, M.J.: Predicting the drape of woven cloth using interacting particles, Computer Graphics (Proc. SIGGRAPH), pp. 365-372, 1994.

[4] Bridson R., Marino S., Fedkiw R.P.: Simulation of clothing with folds and wrinkles, In Symposium on Computer Animation, 2003.

[5] Choi, K.-J., Ko H.-S.: Stable but responsive cloth, Computer Graphics (Proc. SIGGRAPH), 2002.

[6] Cordier, F., Seo H., Magnenat-Thalmann N.: Made-to-measure technologies for online clothing store", IEEE Computer Graphics and Applications, pp. 38-48, January / February 2003.

[7] Desbrun, M., Schroder P., Barr A.: Interactive animation of structured deformable objects, Proceedings of Graphics Interface '99, 1-8, June 1999.

[8] Eberhardt, B., Etzmuss O., Hauth M.: Implicit-explicit schemes for fast animation with particle systems, Eurographics Computer Animation and Simulation Workshop, 2000.

[9] Etzmuss, O., Gross, J., and Straßer W.: Deriving a Particle System from Continuum Mechanics for the Animation of Deformable Objects, IEEE Transaction on Visualization and Computer Graphics, Vol. 9, No. 4, pp. 538-550, 2003.

[10] Eberhardt, B., Weber, A., and Strasser, W. A.: A fast, flexible, particle-system model for cloth draping, IEEE Computer Graphics and Applications 16, pp. 52-59, 1996.

[11] Grinspun, E., Hirani A. N., Desbrun, M., and Schroder, P.: Discrete Shells, In Symposium on Computer Animation, 2003.

[12] Hauth, M. and Etzmuss O.: A High Performance Solver of the Animation of Deformable Objects Using Advanced Numerical Methods, Proc. Eurographics, 2001.

[13] Hauth, M., Etzmuss O., and Straßer W.: Analysis of numerical methods for the simulation of deformable models, The Visual Computer, 2002

[14] House, D. H. and Breen D. E.: Cloth Modeling and Animation, A K Peters, 2000.

[15] Kang, Y.-M. and Cho H.-G.: Complex deformable object in virtual reality, Proc. of Virtual Reality Software and Technology, pp. 49-56, HongKong, Nov 2002.

[16] Müller M., Dorsey J., McMillan L., Jagnow R., Cutler B.: Stable Real-Time Deformations, In Symposium on Computer Animation, pp 49-54, 2002.

[17] Oh, S.-W., Kim H.-S., Wohn K.-Y.: Collision handling for interactive garment simulation, Proc. of VSMM 2002, pp. 239-251, 2002.

[18] Parks, D. and Forsyth D.A.: Improved integration for cloth simulation, EUROGRAPHICS Short Presentation, 2002.

[19] Provot, X.: Deformation constraints in a mass-spring model to describe rigid cloth behavior, Proc. Graphics Interface, pp. 147-154, 1995.

[20] Volino, P., Magnenat-Thalmann N.: Implementing fast cloth simulation with collision response, Proceedings of CGI'00, 2000.

[21] Volino, P., Magnenat-Thalmann N.: Comparing efficiency of integration methods for cloth animation", Proceedings of CGI'01, July 2001.

[22] Vassilev, T., Spanlang B., Chrysanthou Y.: Fast cloth animation on walking avatars, EUROGRAPHICS, 20(3) , 2001.

[23] KAIST Virtual Fitting Room : http://vr.kaist.ac.kr/~redmong/research.htm

**Figure 6:** *Snapshots from dance and ballet animations reproduced by our simulator*