Poster Proceedings of Pacific Graphics (2008)
T. Igarashi, N. Max, and F. Sillion (Editors)

A Simple and Effective Model for Interactive Paper Folding

Young-Min Kang[1†] and Hwan-Gue Cho[2‡]

[1]Dept. of Game Engineering, Tongmyong University, Korea
[2]Dept. of Gomputer Engineering, Pusan National University, Korea

Abstract

Although lots of paper objects are commonly available in everyday life, little efforts have been made to represent the behavior of paper objects in realtime or interactive virtual environments. In this paper, we propose a simple and effective breakable hinge spring model to represent realistic paper appearance and behaviors for interactive applications. The breakable hinge spring model is an adaptive mass-spring mesh structure. In this model, each spring can be broken when its length is shortened by external or internal forces, and the mesh structure is adaptively modified in order to maintain the mass-spring mesh to be a triangular mesh. A stochastic approach was also devised to represent realistic wrinkles on the virtual paper objects. The experimental results show that the proposed method produces realistic paper objects in interactive environments.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensinal Graphics and Realism

1. Introduction

Although paper is a common object in everyday life, there have been little efforts to animate paper objects in interactive or real-time VR environments. In this paper, we propose a simple and effective method for the interactive animation of paper objects in VR or game systems. In order to represent the behavior of a paper object, the well-known mass-spring model can be easily employed. The mass-spring model is the most intuitive and simple representation of deformable soft objects. In the computer graphics society, the animation of soft objects has been intensively studied for a couple of decades.

However, animating paper with mass-spring model leads to a serious numerical problem. The properties of paper require the representing mass-spring model to be extremely stiff, and the stiffness causes numerical instability.

The interactive animation of paper can be described as an efficient numerical integration of motion equation for the damage preserving stiff mass-spring model. This problem is composed of two main issues: 1) stable integration of a stiff ODE (ordinary differential equation), and 2) mesh management for damage preservation.

In the early stage of deformable object animation, geometric approaches have been widely employed because physical-based models require too heavy computation [NG96]. Terzopoulos *et al.* characterized the soft object animation as a deformable surface problem [TPBF87], and their work formed the foundation for the subsequent physics-based approaches.

Since Terzopoulos *et al.* represented soft objects as deformable surfaces [TPBF87], various physically-based approaches have been proposed for soft object animation. Thalmann's group has proposed many techniques for dressed virtual characters [VCMT95]. However, their major interest was the realism of the virtual cloth instead of computational efficiency. Therefore, most of their work employed explicit integration and interactive or realtime animation was impossible.

Various techniques have been proposed for efficient animation of soft objects, and one of the most important advances is the use of implicit integration which guarantees the stability of the animation [BW98]. However, the implicit

† Assistant Professor in Tongmyong University
‡ Professor in Pusan National University

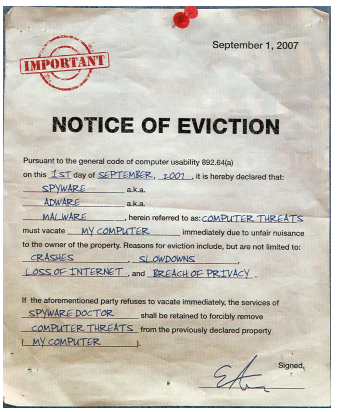


Figure 1: *Peculiar appearance of a typical paper object*

integration involves large linear system [DSB99]. Therefore, some efficient approximate approaches have been proposed [MDDB01], and a stable method that approximates the solution with a direct update formula was also introduced [KC04].

A method to generate seams and wrinkles for virtual cloth was also proposed [MB06]. The method proposed a new model for realistic wrinkles on clothes via seams. This simple model easily incorporates into a mass spring model with improvements over wrinkled appearance. They presented an automated method to construct a seam surface along an arbitrary path on the surface of an irregular mesh.

Although there have been many efforts to reproduce realistic cloth in virtual environments, paper objects cannot be easily animated because wrinkles and crumples on paper objects produces peculiar appearance as shown in Fig. 1.

A discrete shell model was successfully used for describing thin objects [GHDS03]. This model utilizes flexural energy based on the local curvature change and membrane energy based on the stretching. The discrete shell model can efficiently represent the behavior of thin flexible objects such as paper. In this method, however, adaptive mesh restructuring was not taken into account so that crumples can be generated only along the given edges. In order to represent realistic virtual paper, crumples must be generated at arbitrary positions and in arbitrary directions. Therefore, adaptive mesh techniques have to be taken into account.

2. Stable Animation of Stiff Mass-spring

The stability is essential for the real-time or interactive animation of mass-spring model, and implicit integration is the common solution to the stability problem. The simplest implicit method is the backward Euler method, which can be described as follows:

$$\begin{pmatrix} \mathbf{v}^{t+h} \\ \mathbf{x}^{t+h} \end{pmatrix} = \begin{pmatrix} \mathbf{v}^{t} + h\mathbf{M}^{-1}\mathbf{f}^{t+h} \\ \mathbf{x}^{t} + h\mathbf{v}^{t+h} \end{pmatrix} \tag{1}$$

where h denotes the time interval, $\mathbf{v}$ the vector of velocity values of the mass-points, $\mathbf{f}$ the vector of forces, $\mathbf{x}$ the vector of locations of the mass-points, and $\mathbf{M}$ the mass matrix.

Although the implicit method described in Eq.1 seems simple, real-time animation is not an easy problem. The difficulty arises from the unknown force vector at time $t+h$. Since Hooke's law can compute only $\mathbf{f}^t$, the force at the next time step should be approximated by applying Taylor expansion as follows:

$$\mathbf{f}^{t+h} = \mathbf{f}^t + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Delta\mathbf{x}^{t+h} = \mathbf{f}^t + \mathbf{J}\Delta\mathbf{x}^{t+h} \tag{2}$$

where $\mathbf{J}$ denotes the Jacobian matrix of the force vector with respect to the position vector.

By using the force shown in Eq.2, an animation step of the mass spring model can be represented by a typical $\mathbf{A}x = b$ linear system:

$$(\mathbf{M} - h^2\mathbf{J})\Delta\mathbf{v}^{t+h} = h\mathbf{f}^t + h^2\mathbf{J}\mathbf{v}^t \tag{3}$$

For simplicity, let $\mathbf{W}$ denote the matrix $(\mathbf{M} - h^2\mathbf{J})$. The right side of Eq.3 is the sum of a spring force and viscosity force and can be easily computed without any computational problem. Let $\tilde{\mathbf{f}}_i^t$ denote the sum of all spring forces and viscosity force exerted on the mass-point i, and $\tilde{\mathbf{f}}^t$ be the vector of the forces $[\tilde{\mathbf{f}}_1^t, \tilde{\mathbf{f}}_1^t, \cdots, \tilde{\mathbf{f}}_n^t]^{\mathbf{T}}$. Then the problem shown in Eq.3 is efficiently represented as follows:

$$\mathbf{W}\Delta\mathbf{v}^{t+h} = h\tilde{\mathbf{f}}^t \tag{4}$$

The velocity change of each mass-point can be described as follows:

$$\Delta\mathbf{v}_i^{t+h} = \mathbf{W}_{ii}^{-1}(h\tilde{\mathbf{f}}_i^t + h^2 \sum_{(i,j)\in E} \mathbf{J}_{ij}\Delta\mathbf{v}_j^{t+h}) \tag{5}$$

The update formula for mass-point i shown in Eq.5 depends on the same formula for mass-point j. Therefore, we cannot directly update the velocity change of each mass-point. Our method computes the approximate solution with an iterative update based on Jacobi iteration as follows:

$$\begin{aligned} \Delta\mathbf{v}_i^{t+h^{(0)}} &= \mathbf{W}_{ii}^{-1}h\tilde{\mathbf{f}}_i^t \\ \Delta\mathbf{v}_i^{t+h^{(k+1)}} &= \mathbf{W}_{ii}^{-1}(h\tilde{\mathbf{f}}_i^t + h^2 \sum_{(i,j)\in E} \mathbf{J}_{ij}\Delta\mathbf{v}_j^{t+h^{(k)}}) \end{aligned} \tag{6}$$

where $\Delta\mathbf{v}_i^{(k)}$ denotes the velocity change computed with k iterations.

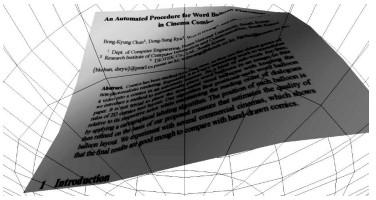


Figure 2: *Rubbery animation of a stiff mass-spring model without breakable springs*

Since $\mathbf{W}$ is diagonally dominant, $\Delta\mathbf{v}^{(0)}$ is already a good approximation, and $\Delta\mathbf{v}^{(1)}$ produced sufficiently stable integration in actual simulation. Fig.2 shows the result of the stable integration described in this paper. However, the resulting animation does not look realistic because no damage is preserved on the surface. In order to effectively represent a paper object, we must also simulate the damages which are not easily recovered by the stiff spring edges.

3. Breakable Hinge Springs

The unsatisfactory result is caused by the static structure of the mass-spring mesh. When real paper is deformed by external forces, the paper is easily damaged and the damage is preserved on the surface. However, the stiff but static mass-spring mesh cannot represent such characteristic.

In order to represent the damage preserving property of paper object, we propose a breakable hinge spring model. The breakable hinge spring model is an adaptive mass-spring mesh structure. In this model, each spring can be broken when its length is shortened by external or internal forces. When a spring is broken, the mesh structure of the paper model should also be modified in order to maintain the mass-spring mesh to be a triangular mesh.

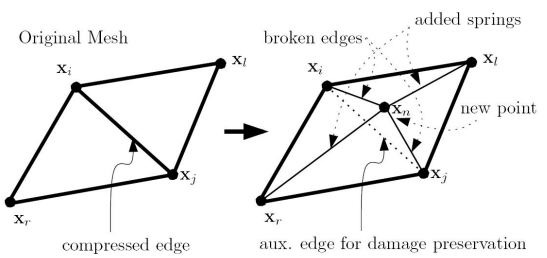
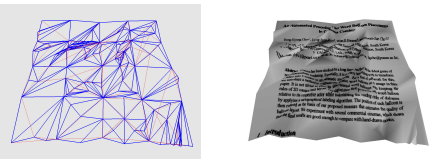


Figure 3: *Breakable spring and mesh adjustment*

Fig.3 illustrates the breakable spring model. On the left side of the figure, a spring edge between two triangles is compressed. The compressed spring is broken and the adjusted mesh structure is shown on the right side. The compressed spring is divided into two distinct spring edges in order to maintain the original length of the compressed spring.



(a) (b)

Figure 4: *Geometric appearance with randomly broken edges: (a) triangular mesh (b) rendered image*

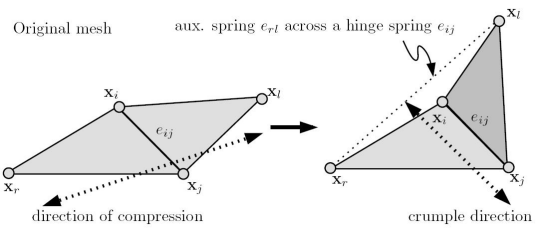


Figure 5: *Auxiliary spring across a hinge folding spring*

The broken spring edge causes a problem in the mesh structure. Because of the broken edge, the mesh is not anymore a triangular mesh. In order to fix the problem, two additional springs are used. After adjusting the mesh structure, the breakable spring model inserts an auxiliary spring to preserve the damage.

In the Fig.4, the auxiliary spring is represented with a dotted line. Because of the auxiliary spring, the mesh remembers the damage and shows the paper-like wrinkles. Fig.4 shows the mesh appearance when springs are randomly chosen and broken. The thick blue lines are breakable springs, and thin red lines are auxiliary spring edges which are not breakable. As shown in the figure the breakable spring model can efficiently represent the damages on the paper surface.

Although the breakable spring model produces paper-like appearance, the method has a severe disadvantage: wrinkles are always across the spring edges. In order to represent the wrinkles along the spring edges, the hinge folding of a spring edge was implemented with an auxiliary spring across the

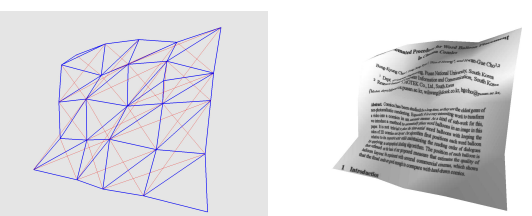


Figure 6: *Geometric appearance with hinge springs*

spring edge as shown in Fig.6. In the figure $\mathbf{x}_i$ and $\mathbf{x}_j$ denote the mass-points linked with a spring, and $\mathbf{x}_r$ and $\mathbf{x}_l$ are the mass-points on the right and the left of the spring respectively. If the distance between $\mathbf{x}_r$ and $\mathbf{x}_l$ is shorter than a given threshold, an auxiliary spring between the mass-points is inserted to the mass-spring structure.

Fig.5 shows the result when auxiliary springs for hinge springs have been inserted. It can be seen that the hinge springs make the surface of the deformable object wavy.

4. Interactive Animation of Realistic Paper Objects

In order to express the appropriate response of paper to the external forces, a stochastic edge break model was devised. The stochastic model determines the fracture probability of each edge. Along with the plausible behavior of the paper object, the realistic animation also requires plausible appearance which can be obtained with complex geometry. For the realistic appearance in real-time environments, we propose a controllable subdivision scheme.

4.1. Stochastic Edge Break for Interactive Animation

In order to produce a plausible animation sequence, an automatic edge selection strategy for breaking or folding springs is necessary. Let $\mathbf{x}_i$ and $\mathbf{x}_j$ be the mass-points linked with a breakable spring. The locations of the mass-points at the original rest state are denoted as $\mathbf{x}_i^0$ and $\mathbf{x}_j^0$, and the locations at the current time t are $\mathbf{x}_i^t$ and $\mathbf{x}_j^t$. The rest length of the spring l_{ij}^0 is, therefore, $|\mathbf{x}_i^0 - \mathbf{x}_j^0|$. The current length of the spring l_{ij}^t is, similarly, $|\mathbf{x}_i^t - \mathbf{x}_j^t|$. Because the spring is broken when it is compressed, the ratio l_{ij}^t / l_{ij}^0 is the most important value for determining whether the spring will be broken or not.

A simple approach is to break the spring according to the contraction ratio $1 - l_{ij}^t / l_{ij}^0$. In our approach, the fracture of an edge actually occurs stochastically. Therefore, we used the ratio as the parameter for computing the probability of the catastrophic phenomena in the mesh structure. In other words, the ratio ranges [1,0] for a compressed spring, and the probability of breaking the spring $\mathbf{P}_{ij}$ is proportional to the contraction ratio as follows:

$$\mathbf{P}_{ij} \propto (1 - l_{ij}^t / l_{ij}^0) \tag{7}$$

Although the contraction of an edge is an important factor for edge break, edges should show different tolerance according to the curvatures along and across the edge. Therefore, we newly propose a stochastic approach based on fragility model which can be computed by curvatures along and across the edge. Let us denote the curvatures along and across an edge as vertical curvature and horizonal curvature respectively. With simple observation, we can easily find that the edge tolerance to the fracture is large when the vertical curvature is small and horizonal one is large.

In order to take the horizontal and vertical curvatures into account, we consider two neighbor vertices along the edge. The neighbor vertices can be seen in Fig.5. In the figure, the neighboring vertices are $\mathbf{x}_r$ and $\mathbf{x}_l$.

The distance between the neighbors, l_{rl}, is $|\mathbf{x}_r - \mathbf{x}_l|$. We denote the normal vectors at the mass-point $\mathbf{x}_i$, $\mathbf{x}_j$, $\mathbf{x}_r$, $\mathbf{x}_l$ as $\mathbf{n}_i$, $\mathbf{n}_j$, $\mathbf{n}_r$, $\mathbf{n}_l$. In our model, the vertical and horizontal curvatures are approximated with dot products of the normal vectors. n_{ij} denotes the dot product of $\mathbf{n}_i$ and $\mathbf{n}_j$, i.e., $\mathbf{n}_i \cdot \mathbf{n}_j$. $n_r l$ is $\mathbf{n}_r \cdot \mathbf{n}_l$ similarly.

When an edge has higher curvature along the edge, the probability of the fracture becomes larger. Contrarily, higher curvature across the edge decrease the probability. Therefore, we can model the fracture probability to be proportional to $(1 - n_{ij})/2$ and $(1 + n_{rl})/2$ as follows:

$$\mathbf{P}_{ij} \propto \left(\frac{1 - n_{ij}}{2}\right)\left(\frac{1 + n_{rl}}{2}\right) \tag{8}$$

It is certain that the probabilty of fracture is proportional to the contraction ratio as shown in Eq. 7. We can also easily observe that the distance between two mass-points linked with an edge (l_{ij}) becomes longer, the probability of fracture should increased. Therefore, the probability can be also modeled to be proportional to $l_{ij}^t / (l_{ij}^t + l_{rl}^t)$. Therefore, the Eq. 8 can be modified as follows:

$$\mathbf{P}_{ij} \propto \left(1 - \frac{l_{ij}^t}{l_{ij}^0}\right)\left(\frac{l_{ij}^t}{l_{ij}^t + l_{rl}^t}\right)\left(\frac{1 - n_{ij}}{2}\right)\left(\frac{1 + n_{rl}}{2}\right) \tag{9}$$

Note that the model shown in Eq. 9 cannot deform any flat objects because the probability is always 0 when the linked mass-points have equivalent normal vectors. Therefore, we employed fragility control parameter ϕ to scale the dot product of the normal vectors. By controlling this parameter, we can change the fragility of a virtual paper object. In our proposed method, the probability of fracture was actually computed as follows:

$$\mathbf{P}_{ij} = \left(1 - \frac{l_{ij}^t}{l_{ij}^0}\right)\left(\frac{l_{ij}^t}{l_{ij}^t + l_{rl}^t}\right)\left(\frac{(1 - \phi n_{ij})(1 + \phi n_{rl})}{4}\right) \tag{10}$$

where ϕ is a parameter that controls the fragility of the springs.

When a spring is broken, a new mass-point is inserted, and the location of the mass-point must be determined. An easy method is to set the location of the new mass-point as the middle of the two mass-point linked with the original spring. However, this simple method recursively breaks the newly inserted broken springs. We adjust the location of the mass-point along the surface normal vector. The magnitude of the adjustment movement h can be easily computed as

follows:

$$h = \frac{\psi}{2}\sqrt{(l_{ij}^0)^2 - (l_{ij}^t)^2} \tag{11}$$

where ψ is a parameter that controls length of broken edges. If ψ is 1, the lengths of the broken edges are exactly the same as the rest length, and the edge would not be recursively broken immediately. However, the edge will tend to be recursively broken when we decrease the parameter ψ. The h is needed only when l_{ij}^t is less than l_{ij}^0 and the spring has been broken.

4.2. Inverse Operation for Broken Edge Recovery

In some cases, some broken edges have to be recovered. Consider the situation shown in Fig. 7 (a). An edge e_{ij} was broken into two difference spring e_{in} and e_{nj}, and e_{ij} became an unbreakable spring, i.e., $e_{ij\cdot breakable} = false$. The unbreakable auxiliary spring contains pointers to newly inserted vertex $\mathbf{x}_n$, and newly added edges e_{in}, e_{nj}, e_{rn}, and e_{nl}. When we need to recover the crumple across the broken edge e_{ij}, we can simply change the breakability property the edge, i.e., $e_{ij\cdot breakable} \leftarrow true$, and remove one vertex $\mathbf{x}_n$ and four edges, e_{in}, e_{nj}, e_{rn}, and e_{nl}. More complicated situation is shown in Fig. 7 (b). In this figure, the broken edge e_{ij} was broken into e_{in}, and e_{nj}, and the edge e_{in} was broken again. Therefore, when we recover a broken edge, we should check whether the edge has been broken into more than two pieces. In our method, we only recover edges broken into only two springs. The inverse operation for broken edge recovery can be described as follows:

if $(e_{ij\cdot breakable} \vee \neg(e_{in\cdot breakable} \wedge e_{nj\cdot breakable}))$ stop

$e_{ij\cdot breakable} \leftarrow true$

$removeEdge(e_{in}, e_{nj}, e_{rn}, e_{nl})$

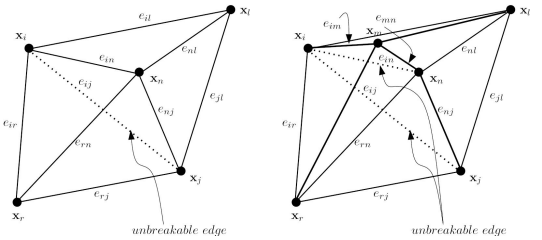
$removeVertex(\mathbf{x}_n)$

4.3. Realistic Visualization of Paper Mesh

In order to represent realistic appearance, complex geometry is required. However, the physical simulation with complex geometry requires heavy computational cost. In our method, the paper objects were modeled with relatively low geometric complexity, but we obtained realistic appearance by subdivision. The original mesh for the physical simulation shown in Fig. 8 (a) is finally rendered after the subdivision as shown in Fig. 8 (b).

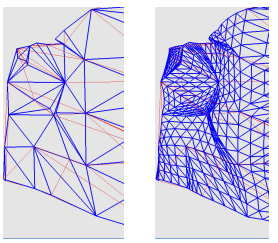
The rendering with subdivision was obtained by recursive function that renders a triangle *DrawTriangle*. Three vertices $(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)$ and three normal vectors $(\mathbf{n}_i, \mathbf{n}_j, \mathbf{n}_k)$ are passed to the function. The function computes the three internal points $m\mathbf{x}_{ij}, m\mathbf{x}_{jk}, m\mathbf{x}_{ki}$, and three internal normal vectors $m\mathbf{n}_{ij}, m\mathbf{n}_{jk}, m\mathbf{n}_{ki}$, and recursively calls the following functions:

$DrawTriangle(\mathbf{x}_i, m\mathbf{x}_{ij}, m\mathbf{x}_{ki}, \mathbf{n}_i, m\mathbf{n}_{ij}, m\mathbf{n}_{ki})$



(a) (b)

Figure 7: *Different situations of broken edges: (a) e_{ij} can be immediately recovered (b) e_{ij} cannot be recovered unless e_{in} is recovered*



(a) (b)

Figure 8: *Subdivision for fast and realistic visualization: (a) original mesh (b) subdivided mesh*

$DrawTriangle(m\mathbf{x}_{ij}, \mathbf{x}_j, m\mathbf{x}_{jk}, m\mathbf{n}_{ij}, \mathbf{n}_j, m\mathbf{n}_{jk})$

$DrawTriangle(m\mathbf{x}_{ij}, m\mathbf{x}_{jk}, m\mathbf{x}_{ki}, m\mathbf{n}_{ij}, m\mathbf{n}_{jk}, m\mathbf{n}_{ki})$

$DrawTriangle(m\mathbf{x}_{jk}, \mathbf{x}_k, m\mathbf{x}_{ki}, m\mathbf{n}_{jk}, \mathbf{n}_k, m\mathbf{n}_{ki})$.

Since the crumples on the paper surface should not be smoothed, we computed each internal point by considering only the end points and the normal vectors as shown in Fig.9. Assuming that we have to compute an internal point $m\mathbf{x}_ij$ by considering the end points $\mathbf{x}_i$ and $\mathbf{x}_j$, and normal vectors $\mathbf{n}_i$ and $\mathbf{n}_j$. Let us consider a smooth curve $C(u)$ that starts at $C(0) = \mathbf{x}_i$ and ends at $C(1) = \mathbf{x}_j$. The internal point $m\mathbf{x}_{ij}$ must be on the curve, and actually $C(0.5)$ can be regarded the internal point. The smooth curve can easily obtained by employing *Hermite* interpolation. The *Hermite* curve requires the derivatives at the end point, i.e., $C'(0)$ and $C'(1)$. Because the derivatives must be perpendicular to the normal

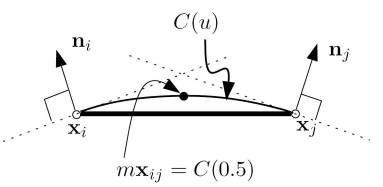
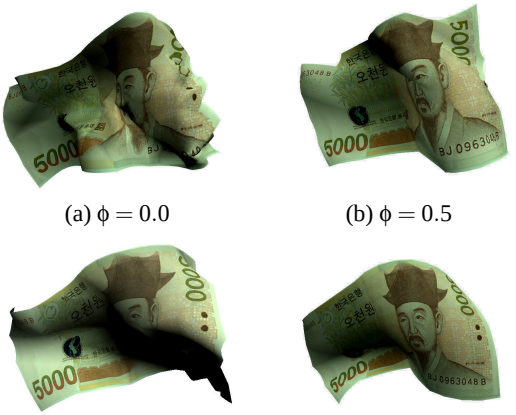


Figure 9: *Internal point $m\mathbf{x}_{ij}$ for mesh subdivision*



(a) $\phi = 0.0$ (b) $\phi = 0.5$

(c) $\phi = 0.75$ (d) $\phi = 1.0$

Figure 10: *Effect of control parameter ϕ while ψ is fixed as 1.0: (a) $\phi = 0.0$, (b) $\phi = 0.5$, (c) $\phi = 0.75$, and (d) $\phi = 1.0$*

vectors, we can easily compute the derivatives at the starting and ending points by projecting the vector from $\mathbf{x}_i$ to $\mathbf{x}_j$ onto the plane perpendicular to the normal vectors at the terminal points. The internal point, then, can be easily computed as follows:

$$C(0) \leftarrow \mathbf{x}_i$$
$$C1 \leftarrow \mathbf{x}_j$$
$$\mathbf{v} \leftarrow \sigma(\mathbf{x}_j - \mathbf{x}_i)$$
$$C'(0) \leftarrow \mathbf{v} - (\mathbf{v} \cdot \mathbf{n}_i)\mathbf{n}_i$$
$$C'(1) \leftarrow \mathbf{v} - (\mathbf{v} \cdot \mathbf{n}_j)\mathbf{n}_j$$
$$m\mathbf{x}_{ij} = C(0.5) \leftarrow \frac{\mathbf{x}_i + \mathbf{x}_j}{2} + \frac{C'(0) - C'(1)}{8}$$

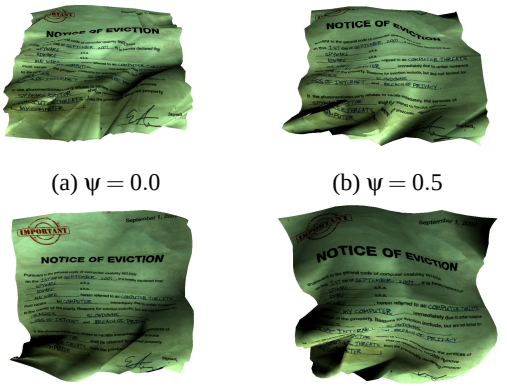
where σ controls the sharpness of the paper crumples.

5. Experiments

The control parameters for the proposed method are ϕ, ψ, and σ. The parameter ϕ controls the fragility of paper objects while ψ controls the tendency of recursive edge break. The parameter σ does not controls the physical simulation. σ controls the smoothness of the subdivided mesh for the final rendering. According to the control parameters, the appearance and the behavior of the paper model is changed.

Fig.10 shows the effect of the control parameter ϕ. As shown in the figure, a larger ϕ value generates stiffer paper behaviors while a lower value produces softer paper deformation. Fig.11 shows the effect of the control parameter ψ. Because an edge tends to recursively breaks into smaller pieces when ψ is small, the parameter ψ could be successfully used for controlling the crumples on the paper surface.

Fig.12 shows the effect of the control parameter σ. The



(a) $\psi = 0.0$ (b) $\psi = 0.5$

(c) $\psi = 0.75$ (d) $\psi = 1.0$

Figure 11: *Effect of control parameter ψ while ϕ is fixed as 0.5: (a) $\psi = 0.0$, (b) $\psi = 0.25$, (c) $\psi = 0.75$, and (d) $\psi = 1.0$*

(a) $\sigma = 0.0$ (b) $\sigma = 0.5$ (c) $\sigma = 1.0$

Figure 12: *Effect of control parameter σ ($\phi = 0.5$, and $\psi = 0.25$: (a) $\sigma = 0.0$, (b) $\sigma = 0.5$, and (c) $\sigma = 1.0$)*

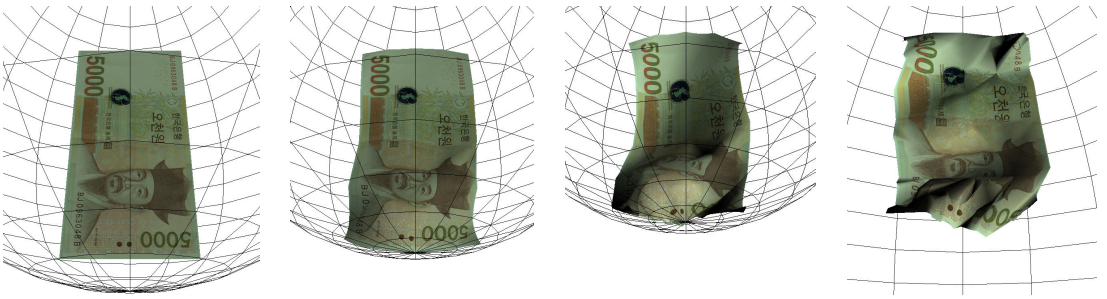
parameter σ controls smoothness of the subdivided mesh. As shown in the figure, the surface becomes smoother as the parameter increases.

Fig.13 shows the animation result when breakable hinge springs are employed in interactive environments. The springs are broken or folded according to the external force exerted by the sphere. As shown in the figure, the breakable hinge spring model can produce plausible paper animation. The animation shown in Fig. 13 required 494 vertices and 3856 spring edges (only the vertices and springs for physical simulation were counted). We tested the interactive performance of the proposed method on common personal computing environments such as desktop PC with 3.00 GHz Pentium 4 CPU and without any hardware acceleration. In our test for generating the result shown in Fig. 13, the animation performance was 58.8 frames per second on average.

6. Conclusion

In this paper, we proposed a simple and effective approaches for efficient animation of paper objects in interactive environments such as VR systems and games.

The proposed method is intuitive because it is based on a simple and effective mass-spring model and uses a direct update formula instead of solving the large linear system.



(a) (b) (c) (d)

Figure 13: *Interactive animation of the paper model with parameters* $\phi = 0.1$, $\psi = 0.25$, *and* $\sigma = 1.0$. *The animation performance was 58.8 fps with 494 vertices and 3856 springs (only the vertices and springs for physical simulation were counted)*

Therefore, the method produces stable animation of stiff mass-spring model.

In order to obtain the peculiar appearance and the behavior of paper object, we proposed a method for generating and preserving crumples on a paper surface: *breakable hinge spring model*. Although paper objects are easily deformed by external forces like cloth objects, the deformed paper surface is not easily recovered while the wrinkles on cloth objects are immediately vanishes. In order to tackle this problem, we proposed the breakable hinge spring model. The breakable hinge spring model is an adaptive mass-spring mesh structure. In this model, each spring can be broken when its length is shortened by external or internal forces. Although the contraction of an edge is an important factor for edge break, edges should show different tolerance according to the curvatures along and across the edge. Therefore, we proposed a stochastic approach based on fragility model which can be computed by curvatures along and across the edge.

In order to achieve realistic appearance, we subdivided the original mesh used for physical simulation with a controllable parameter σ, and the experimental results show that we can efficiently visualize a plausible paper object in virtual worlds.

The experimental results show that the proposed method produces realistic paper objects in interactive environments, and the method can be successfully utilized in real-time or interactive games or VR environments.

Acknowledgment

This research was supported by CRC research grant 1-07-1205-006-10010-00-001 from Korea Culture & Content Agency(KOCCA).

References

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. *Proceedings of SIGGRAPH 98* (July 1998), 43–54. 1

[DSB99] DESBRUN M., SCHRÖDER P., BARR A.: Interactive animation of structured deformable objects. *Graphics Interface '99* (June 1999), 1–8. 2

[GHDS03] GRINSPUN E., HIRANI A., DESBRUN M., SCHRODER P.: Discrete Shells. In *ACM SIGGRAPH / Eurographics Symposium on Computer Animation* (Aug 2003), pp. 62–67. 2

[KC04] KANG Y.-M., CHO H.-G.: Real-time animation of complex virtual cloth with physical plausibility and numerical stability. *Presence - Teleoperators and Virtual Environments 13*, 6 (2004), 668–680. 2

[MB06] MA L., BACIU J. H. G.: Generating seams and wrinkles for virtual clothing. *Proceedings of ACM International Conference on Virtual Reality Continuum and Its Applications* (2006), 14–17. 2

[MDDB01] MEYER M., DEBUNNE G., DESBRUN M., BAR A. H.: Interactive animation of cloth-like objects in virtual reality. *The Journal of Visulalization and Computer Animation 12* (2001), 1–12. 2

[NG96] NG H. N., GRIMSDALE R. L.: Computer graphics techniques for modeling cloth. *IEEE Computer Graphics & Applications 16*, 5 (September 1996), 28–41. 1

[TPBF87] TERZOPOULOS D., PLATT J., BARR A., FLEISCHER K.: Elastically deformable models. *Computer Graphics (Proceedings of SIGGRAPH 87) 21*, 4 (July 1987), 205–214. 1

[VCMT95] VOLINO P., COURSHESNES M., MAGNENAT-THALMANN N.: Versatile and efficient techniques for simulating cloth and other deformable objects. *Proceedings of SIGGRAPH 95* (August 1995), 137–144. 1