An Oriented Particle and Generalized Spring Model for Fast Prototyping Deformable Objects

Il-Kwon Jeong and Inho Lee

Digital Actor Research Team, Electronics and Telecommunications Research Institute, Korea

Abstract

We present a new mass-spring system, in particular, by using an oriented particle and generalized spring model, which can be used for fast prototyping or animation of deformable objects. Conventional mass-spring system is widely used especially in cloth animation. However animating deformable objects such as a jelly cube requires extra diagonal springs in addition to structural springs in order to provide shear strain force and guarantee stability on the original shape formation. One has to use his a priori knowledge or trial-and-error method to construct a stable spring network for a given deformable object. This is due to the inherent one-dimensional nature of the conventional spring model. In order to overcome the difficulty in designing a spring network and make it possible to construct a stable network easily and intuitively, we propose an oriented particle and generalized spring model that reacts against bending and twisting force as well as stretching. By using our new mass-spring system, one can easily construct a spring network from a given geometric model. Moreover, one can animate one-dimensional flexible object such as a mobile as well as a thin two-dimensional object with sharp folds or creases. In addition, we present an offsetting method for tweaking mass-spring system to have valuable properties that help animating deformable objects with various shapes.

Categories and Subject Descriptors: I.3.5 [Computer Graphics]: Physically based modeling

1. Introduction

The computer graphics community have been especially interested in animating cloth like objects among the various deformable objects. Researchers became interested in cloth modeling in the late 1980s while the beginning of modern cloth modeling research is traced back to the 1930s [HB00].

There are increasing demands for producing physically based animations than ever owing to the advances in computing hardware. Physically based simulation engine for 3D games is a good example for such a trend. A physically based simulation can add realism to graphic content and make a user interact with the content in a realistic way.

Previous works on deformable body animation can be categorized into two approaches; geometrical or mathematical techniques and dynamic simulation. Our concern is about the dynamic simulation of deformable objects.

Conventional mass-spring system is widely used in deformable body animation. However animating deformable objects such as a jelly cube requires extra diagonal springs in addition to structural springs in order to provide shear strain force and guarantee stability on the original shape formation. One has to use a priori knowledge or trial-and-error method to construct a stable spring network for a given deformable object. This is due to the inherent one-dimensional nature of the conventional spring model.

In order to overcome the difficulty in designing a stable spring network and make it possible to construct a stable network easily and intuitively, an oriented particle and generalized spring model that reacts against bending and twisting force as well as stretching is proposed. By using the new mass-spring system, one can easily construct a spring network from a given geometric model. Moreover, one can animate one-dimensional flexible object such as a mobile and a very thin two-dimensional object with sharp folds or creases.

2. Previous work

Previous works can be categorized as two approaches. They are geometric techniques and dynamic simulation. Because our concern is the latter, we will discuss works

that describe the dynamic simulation for deformable objects or cloth.

Many works have been done to achieve real time simulation. Baraff et al. used the implicit Euler integration method such that large time steps can be used [BW98]. Desbrun et al. split the forces in mass-spring system into linear and non-linear parts. The non-linear part is first neglected to rapidly approximate the implicit integration and then the estimate is corrected to preserve momentum [DSB99]. Kang et al. improved this method with an approximate implicit integration and they also improved the appearance of the cloth by tessellating the triangle and introducing a wrinkled cubic spline curve [KCCL01]. Choi and Ko proposed a semi-implicit cloth simulation technique that is very stable and responsive to produce a very realistic-looking cloth [CK02]. Cordier and Thalmann proposed a hybrid method that use several different algorithms for the pieces of a garment for a real-time animation of a fully dressed virtual human [CT02]. However, all these methods work on a conventional mass-spring system. So the demerits of the conventional system still remain.

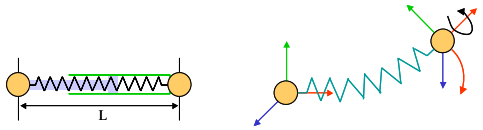
Sederberg and Parry introduced the use of free-form deformations(FFDs) as an efficient method for aimating soft bodies via a structural hyperpatch [SP86][CHP89]. Lander proposed an idea that constructing a mass-spring system for those CVs [Lan00]. However, this approach is not good for an interactive simulation because it is hard to define the realationship between an exerted external force and the CVs due to the ambiguity in matching between the CVs and the vertex of model being affected by the force. Müller et al. proposed a stiffness warping technique for real-time deformation based on Green's non-linear strain tensor [MDM*02]. Though they suggested a relatively fast and accurate method applicable to both FEM and mass-spring systems, the resulting algorithm is complicated due to the inherent complexity in elasticity-based methods. Oshita and Makinouchi presented a method combining dynamic simulation and geometric techniques [OM01]. However, their normal control mechanism is complicated and is not intuitive.

For shape design and rapid prototyping applications, Szeliski and Tonnesen developed oriented particles that are distributed model of surface shape [ST92]. However, they have dealt with separated particles and not linked particles and the potential functions used are heuristic. House and Breen suggested the use of three kinds of cloth springs, separation, bending, and trellising. Because they are imaginary springs, it is far from a physically proper dynamic simulation model, and it only uses the position of the masses, so a rumple or creases in clothes cannot be animated.

The authors proposed a 3D spring that exhibits a bending resisting force by using an oriented mass with unique direction vector for each attached spring [JL03]. However, 3D spring does not create any restoring force or torque against twisting. Because a properly designed conventional mass-spring model exhibits a twisting resisting force as well as a bending resisting force, the 3D spring model is not sufficient for general deformable body simulation.

3. Problems in previous mass-spring system

In order to simulate cloth with conventional mass-spring system one need to include three types of springs. These are structural springs, shear springs, and bend (flexion) springs. The shear springs prevent the model from being stretched unacceptably in diagonal directions and the bend springs exert forces against a bending (folding) of the model. However, adding two extra types of springs in addition to the structural springs is not intuitive. The need for the extra types of springs originates from the linear (1 dimensional) nature of the conventional spring. Figure 1 shows schematic diagrams comparing a spring in conventional mass-spring system and one in a real world.



(a) Linear spring (b) Real spring

Figure 1: *Linear spring and real spring models*

Figure 2 shows an intuitive conventional spring model for a jelly cube that may collapse and a stable one with extra diagonal springs.

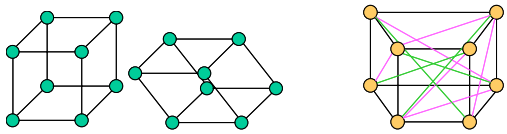


Figure 2: *An intuitive unstable conventional spring models for a jelly cube and a stable one with extra springs*

3D spring proposed in [JL03] cannot handle a model shown in Figure 3 due to lack of twist restoring force (circled arrow) shown in Figure 1 (b).

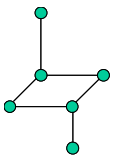


Figure 3: *A mobile model*

4. Oriented particle and generalized spring model

This section describes our oriented particle and generalized spring model.

4.1. Idea

In order to make a spring model have a twist-resisting property, we extend the 3D spring model to have additional reference vector for twist. Detailed explanations

and equations of motion are derived in the next section, and Figure 4 shows a schematic diagram showing the mechanism of the proposed generalized spring model. Where the box represents a particle. Reference bending vector and bending restoring torque (in blue), and twist vector and twist restoring torque (in red) are shown. The bending- and twist-restoring torques are calculated using angles of rotation between the local coordinates of the respective particles and the reference vectors.

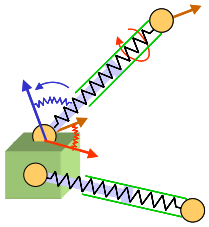


Figure 4: *Schematic diagram of the proposed generalized spring*

4.2. Oriented particle and generalized spring model

Figure 5 shows the proposed oriented particle and generalized spring model. The two particle, A and B, are connected by a generalized spring.

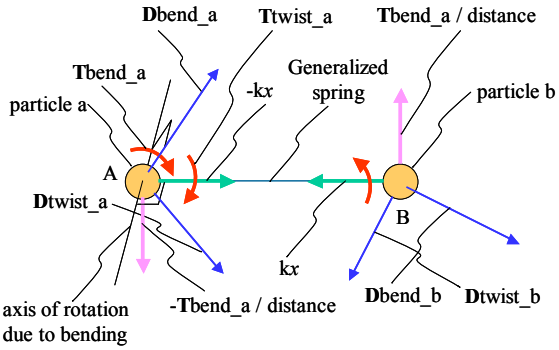


Figure 5: *Oriented particle and generalized spring model*

For the restoring forces, each particle has two reference vectors for each spring connected to it. **D**bend_a means a reference bending vector for particle A and **D**twist_a is a reference twist vector for particle A. All reference vectors are assumed to be normalized. The force or torque acting via the generalized spring consists of three components: the expansion force owing to the unique property of the one dimensional spring, the bending-restoring torque and the twist-restoring torque. Because a negative expansion force means a contraction force as a result, hereinafter the expansion force will be described without distinction from the contraction force. The bending-restoring torque is based upon the angle difference between a reference bending vector of a mass and a spring direction vector. The twist-restoring torques is based upon the angle difference between the two projected reference twist vectors of the particles onto a plane normal to the spring length vector.

Listing 1: *Pseudo code for calculating the spring forces and torques*

```
Function [Fa, Ta, Fb, Tb] = spring(Pa, Va, Dbend_a,
Dtwist_a, Wa, Pb, Vb, Dbend_b, Dtwist_b, Wb, Kp,
Jb_p, Jt_p, L)

vecAB = Pb – Pa // vector from particle A to B
d = norm(vecAB) // distance between A and B
vecAB = normalize(vecAB)

θ = acos (Dbend_a•vecAB) // angle between two vectors
Tbend_a = normalize(Dbend_a × vecAB)*Jb_p*θ
          – damping(Wa) // bending restoring torque A
Fb = normalize(Dbend_a – vecAB*  // force acting on B
     (Dbend_a•vecAB) )*Kp*θ / d // due to Tbend_a
Fa = –Fb // reaction force of the above
vecX = normalize(Dtwist_a-vecAB*(Dtwist_a•vecAB))
vecY = normalize(Dtwist_b-vecBA*(Dtwist_b•vecBA))
θ = acos (vecX• vecY) // angle between twist vectors
Ttwist_a = normalize(vecX × vecY)*Jt_p*θ
           – damping(Wa) // twist restoring torque on A
Ttwist_b = -Ttwist_a

Fa = Fa + vecAB*(d – L)*Kp // expansion force on A
Fb = Fb – vecAB*(d – L)*Kp // expansion force on B

θ = acos (Dbend_b•vecBA)
Tbend_b = (Dbend_b × vecBA)*Jb_p*θ– damping(Wb)
Ftemp = normalize(Dbend_b – vecBA*(Dbend_b
        •vecBA)) )*Kp*θ / d – damping(Va, Vb)
Fa =  Fa + Ftemp , Fb =  Fb – Ftemp

Ta = Tbend_a + Ttwist_a , Tb = Tbend_b + Ttwist_b
```

Listing 1 shows the pseudo code for calculating the spring forces and torques according to the above explanations. Where **V**a is the velocity vector, **P**a is the position vector, **W**a is the rotational velocity vector, and L is the unstretched spring length. Parameters for particle B can be similarly defined. Spring constant is Kp, restoring torque constant against bending is Jb_p, and restoring torque constant against twisting is Jt_p. Note that reference vectors are attached to a local coordinate of a particle. Numerical integration can be similarly performed as in [JL03].

4.3. Offsetting method

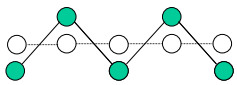


Figure 6: *Offseting particles*

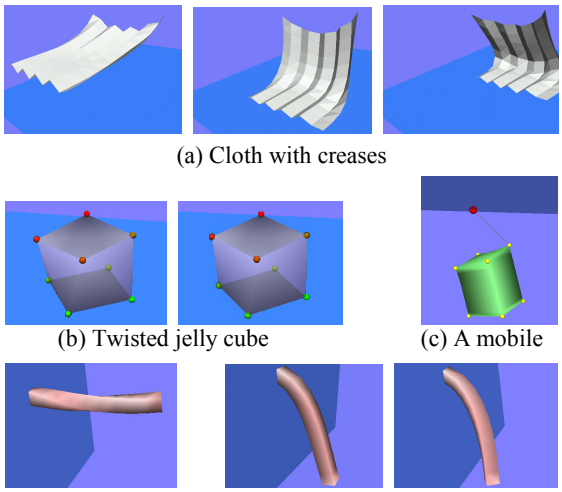
Figure 6 explains offset of particles. White particles are real, and green ones are virtual ones offsetted upward or downward. By simulating with real paticles and rendering using the virtual ones, one can achieve sophisticated deformable body simulation with a relatively simple mass-

spring model. The offset may include translational or rotational transformation.

5. Implementation results

As expected, deformable object created with oriented particle and generalized spring model exhibits robustness to external torsion. Figure 7 shows several simulation results. If no torsional force is applied, generalized spring model operates as the same as 3D spring. A twisted jelly cube with generalized spring more rapidly convrege to its original shape than one with 3D spring.

A bar attached to a wall under gravity shows a noticeable difference between 3D spring and generalized spring. The bar is initially positioned horizontally and twisted by 180 degrees. While a bar constructed with 3D spring remains twisted in its steady state, a bar with generalized spring recovers untwisted configuration.



(a) Cloth with creases

(b) Twisted jelly cube

(c) A mobile

(d) A bar attached to a wall: initial configuration, steady state of generalized spring and 3D spring

Figure 7: *Implmentation results*

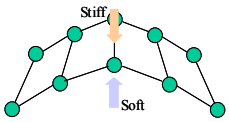


Figure 8: *A model with non-symmetrical resistance*

By intuitively assigning non-linear or non-symmetrical function to restoring torque constants one can achieve non-symmetrical resistance of a deformable object (see Figure 8), which is impossible with the conventional mass-spring system.

6. Conclusions

We extended the 3D spring to propose a generalized spring that exhibits bending and twisting restoring properties. It can be used for fast prototyping deformable object from any given geometric model. The most beneficial feature of generalized spring is that it can simulate one-dimensional structure and thin objects with creases, and can be reverted to linear or 3D springs by simply selecting appropriate spring coefficients. To our knowlege there has been no simple spring model that exhibits the above properties.

References

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *Proc. SIGGRAPH '98* (1998), 43-54.

[CHP89] CHADWICK J.E., HAUMANN D.R., PARENT R.E.: Layered construction for deformable animated characters. In *Proc. SIGGRAPH '89* (1989), 23:243-252.

[CK02] CHOI K.J., KO H.S.: Stable but responsive cloth. In *Proc. SIGGRAPH '02* (2002), 604-611.

[CT02] CORDIER F., THALMANN N.: Real-time animation of dressed virtual human. *Computer Graphics Forum 21*, 3 (2002), 327-335, (Proc. Eurographics '02)

[DSB99] DESBRUN M., SCHRÖDER P., BARR A.: Interactive animation of structured deformable objects. *Proc. of Graphics Interface* (1999), 1-8.

[HB00] HOUSE D.H., BREEN D.E.: *Cloth Modeling and Animation*. A K Peters, Ltd., 2000.

[JL03] JEONG I.K., LEE I.H.: A new 3D spring for deformable object animation, *Eurographics '03 - Interactive demos and posters* (2003), 67-70.

[KCCL01] KANG Y.M., CHOI J.H., CHO H.G., LEE D.H.: An efficient animation of wrinkled cloth with approximate implicit integration. *The Visual Computer*, 17, 3 (2001), 147-157.

[Lan00] LANDER J.: Graphic content - in this corner... the crusher!. *Game Developer Magazine* (June 2000), 17-22.

[MDM*02] MÜLLER M., DORSEY J., MCMILLAN L., JAGNOW R., CUTLER B.: Stable real-time deformations. *Proc. of ACM SIGGRAPH Symposium on Computer Animation* (2002), 49-54.

[OM01] OSHITA M., MAKINOUCHI A.: Real-time cloth simulation with sparse particles and curved faces. *Proc. of Computer Animation* (2001), 220-227.

[SP86] SEDERBURG T.W., PARRY S.R.: Free-form deformation of solid geometric models. In *Proc. SIGGRAPH '86* (1986), 20:151-160.

[ST92] SZELISKI R., TONNESEN D.: Surface modeling with oriented particle systems. In *Proc. SIGGRAPH '92* (1992), 26:185-194.