Commun. Comput. Phys.
doi: 10.4208/cicp.120612.080313a

Vol. **14**, No. 5, pp. 1228-1251
November 2013

Numerical Method of Fabric Dynamics Using Front Tracking and Spring Model

Yan Li[1,*], I-Liang Chern[2,3,4], Joung-Dong Kim[1] and Xiaolin Li[1]

[1] *Department of Applied Mathematics and Statistics, Stony Brook University, Stony Brook, NY 11794-3600, USA.*

[2] *Department of Applied Mathematics, Center of Mathematical Modeling and Scientific Computing, National Chiao Tung University, Hsin Chu, 300, Taiwan.*

[3] *Department of Mathematics, National Taiwan University, Taipei, 106, Taiwan.*

[4] *National Center for Theoretical Sciences, Taipei Office, Taipei, 106, Taiwan.*

Received 12 June 2012; Accepted (in revised version) 8 March 2013

Communicated by Ming-Chih Lai

Available online 13 June 2013

Abstract. We use front tracking data structures and functions to model the dynamic evolution of fabric surface. We represent the fabric surface by a triangulated mesh with preset equilibrium side length. The stretching and wrinkling of the surface are modeled by the mass-spring system. The external driving force is added to the fabric motion through the "Impulse method" which computes the velocity of the point mass by superposition of momentum. The mass-spring system is a nonlinear ODE system. Added by the numerical and computational analysis, we show that the spring system has an upper bound of the eigen frequency. We analyzed the system by considering two spring models and we proved in one case that all eigenvalues are imaginary and there exists an upper bound for the eigen-frequency. This upper bound plays an important role in determining the numerical stability and accuracy of the ODE system. Based on this analysis, we analyzed the numerical accuracy and stability of the nonlinear spring mass system for fabric surface and its tangential and normal motion. We used the fourth order Runge-Kutta method to solve the ODE system and showed that the time step is linearly dependent on the mesh size for the system.

AMS subject classifications: 74B20, 65D17, 65Z05

Key words: Front tracking, spring model, eigen frequency.

[*] Corresponding author. *Email addresses:* `yli@ams.sunysb.edu` (Y. Li), `chern@math.ntu.edu.tw` (I-L. Chern), `selom114@ams.sunysb.edu` (J.-D. Kim), `linli@ams.sunysb.edu` (X. Li)

©2013 Global-Science Press

1 Introduction

Fabric material belongs to flexible objects and is more difficult to model than rigid objects. Accurate coupling of the fabric material with airflow is even more challenging. However modeling of its dynamic motion is demanded in both animation industry and engineering science. A fabric surface can be considered as a membrane which is an idealized two dimensional manifold for which forces needed to bend it are negligible when compared with forces needed to stretch and compress it. For such surface, the spring model on a triangulated mesh is a good mathematical approximation. Simulation of fabric dynamics through computational method has applications in both computer graphics and engineering. The textile and fashion industry invites computer tools that can realistically generate the shape of a cloth dressing. Scientific applications include modeling of cell skin and soft tissues. Our motivation started from the numerical study of the parachute system and its coupling with the airflow.

Many authors have contributed to the modeling of cloth and fabric surface. Terzopoulos and Fleischer [24–26] proposed continuous model for the deformable objects. Aono *et al*. [2,3] used the Tchebychev net cloth model to simulate a sheet of woven cloth composites in which they presented two algorithms, a finite difference method for the Tchebychev net and the algorithm for fitting a given 2D broadcloth composite ply to a given 3D curved surface represented by a NURBS surface. Late in 1990's and 2000's particle method gained popularity due to its intuitiveness and simplicity. Breen *et al*. [9,10], presented a particle-based model capable of being tuned to reproduce the static draping behavior of specific kinds of woven cloth. Eberhardt *et al*. [17,18] extended the model and introduced techniques to model measured force data exactly and thus cloth-specific properties. They also extended the particle system to model air resistance. Choi and Ko [15] also used particle spring model, our model is base on their work.

On the physics based modeling, Platt and Barr [20] showed how to use mathematical constraint methods based on physics and on optimization theory to create controlled, realistic animation of physically-based flexible models. Carignan *et al*. [14] discussed the use of physics-based models for animating clothes on synthetic actors in motion. Provot [22] described a physically-based model for animating cloth objects, derived from elastically deformable models, and improved order to take into account the non-elastic properties of woven fabrics. Volino *et al*. [28] presented an efficient set of techniques that simulates any kind of deformable surface in various mechanical situations. Other applications of spring model include Selle *et al*. [1] in their application to hair modeling, Terzopoulos and Waters [27] on the modeling of cutaneous tissue, subcutaneous tissue and muscle layer, and Terzopoulos *et al*. [26] modeling of deformable curves and surfaces using the elasticity theory. It should be mentioned none of the above papers has discussed the spectrum and upper bound of the frequencies and its crucial role in the oscillatory motion of the spring system.

In this paper, we follow the general idea of the particle and spring mass method for the modeling and simulation of the cloth stretching and draping. Our main focus is on the numerical aspect of the ODE system, its accuracy and stability. For this purpose, we will analyze the oscillatory and non-oscillatory behaviors of the spring system and for the oscillatory motion, we will give a numerically added proof for the upper bound of the eigen frequency. Based on the analysis of eigen frequencies, we studied the numerical stability criterion and accuracy of the numerical schemes for solving such a system.

2 Mathematical model and numerical method

The method we have used in this paper for the simulation of fabric system contains several components. The fabric surface is modeled by a spring system on a homogeneously triangulated mesh. The data structures and many functionalities are based on the *FronTier* library developed for the front tracking software library. The numerical method is a fourth order Runge-Kutta scheme. To simplify the interaction between the fabric surface and the external driving force, we design a special method which separates the impulse from external sources such as gravity and fluid pressure, and the impulse from internal forces each point mass receives from its neighboring points under the spring model.

The resulting equations for the spring mass model is an ODE system. To accurately and efficiently solve this system, it is important to understand the characteristic motion of the mass points. In particular, we need to understand the eigen frequencies of the oscillatory modes and estimate the upper bound of the eigen frequencies. The choice of numerical scheme and the criterion for choosing time step will affect the stability and accuracy of the solution. In this paper, We use the fourth order Runge-Kutta method to solve the spring system. In the following subsections, we will show that with appropriate choosing of the time step based on the upper bound of the eigen frequencies, the explicit scheme is not only stable, but also efficiently and accurately solves the ODE system.

Due to the lack of understanding on the numerical stability of the nonlinear ODE system, many researchers have tried to use implicit method. For example, Baraff *et al*. [5] described a cloth simulation system that can stably take large time steps. It introduces an implicit integration method and applies on a triangulated mesh. The resulting large sparse systems of linear equations are solved by a modified preconditioned conjugate gradient (MPCG) algorithm. Ascher *et al*. [4] improved the robustness and efficiency of the MPCG algorithm [5]. They also proved convergence, which lead to a correction in the initiation stage of the original algorithm with improved efficiency. Although implicit methods can take relatively large step while maintaining numerical stability of the ODE system, a carefully designed high order explicit method is more accurate.

2.1 Mesh representation of fabric surface

Using spring mass system to model a fabric surface has been explored by computer scientists and applied mathematicians. This spring system provides good model for the simulation of thin surfaces such as skin, soft tissue, paper and textile. It has also become a natural choice for the modeling of leaves and parachute canopy. We followed the work

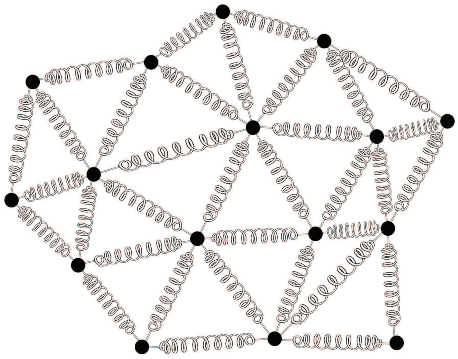

Figure 1: The spring model on a triangulated mesh. Each vertex point in the mesh represents a mass point with point mass m. Each edge of the triangles has a equilibrium length set during initialization and the changing length exerts a spring force on the two neighboring vertices in opposite directions.

by Choi and Ko [15] and applied to the triangular mesh from the front tracking library. Although the basic idea is similar, there are several marked differences in our application.

Front tracking method treats a hyper-surface as a topologically linked set of marker points. This method has been used for the simulations of fluid interface instabilities [19], diesel jet droplet formation [8], and plasma pellet injection process [23]. In these problems, the hyper-surface is used to model the interior discontinuities of materials and such manifold surface may undergo complicated topological changes. Front tracking library contains data structure and functionalities to handle dynamic evolution of surface topology and to optimize hyper-surface mesh consistently. The basic data structure and many functionalities are also used in our fabric modeling and simulation. The evolution of fabric surface is simpler in topological handling due to the fact that a fabric surface can neither bifurcate nor merge. However the fabric system has certain constraints and poses new challenges to the front tracking data structure and some associated functionalities.

The first new requirement is that the hyper-surface area must be conserved. This requirement prompts us to add an equilibrium state of the mesh and treat the marker points of the hyper-surface as a set of spring vertices. The general property of a fabric surface is that it is a non-stretchable surface. However, it is very difficult to numerically maintain a meshed surface with constant area with non-deformable simplexes. Since there is always a finite elasticity of a fabric material, therefore to approximate the fabric surface as a highly stiffened spring system is not only convenient, but also realistic. One of the major revision of the data structure is to add the equilibrium length for each side of the simplexes (which are triangles for 3D hyper-surface). This variable is computed after the initial mesh optimization and stored in memory throughout the computation. Fig. 1 illustrates the spring model on the triangulated mesh using the front tracking data structure.

The second difference between the material interface and the fabric surface lies in the fact that the former is a manifold while the latter is an immersed surface. For the manifold surface, the positive and negative sides are not connected and can always be represented as a level surface with positive and negative level function values on each side. For an immersed surface, although one can still mark its positive and negative sides, such topological orientation is only local. A space point with short distance away from the non-manifold surface cannot be classified as to which side of the hyper-surface it belongs. Such ambiguity poses difficulty in computing interpolation. We tackle this problem by using an index coating method. This method allows us to distinguish which side a space point belongs to on the local basis (a few mesh blocks away from the surface).

Front tracking method also relies on the index of the side to check the topological consistency of the interface. For an immersed interface, there is no topological bifurcation, but there will still be collision of the hyper-surface mesh points. Therefore, new functions to detect and resolve the marker point collision are needed, especially when the fabric surface is folded. In addition, the steady mesh triangulation throughout the computation makes the global indexing of vertices and triangles relatively easy to implement. Such global indexing makes parallel partition and communication more accurate and robust.

2.2 Spring models

When no external driving force is applied, the fabric surface, which is represented by the spring mass system, is a conservative system whose total energy (kinetic energy plus potential energy) is a constant. Assuming each mesh point represents a point mass m in the spring system with position vector $\mathbf{x}_i$, the kinetic energy of the point mass i is $T_i = \frac{1}{2}m|\dot{\mathbf{x}}_i|^2$, where $\dot{\mathbf{x}}_i$ is the time derivative, or velocity vector of the point mass i.

We consider two types of spring systems. The first one, which we will refer to as Model-I, has the potential energy between two point masses $\mathbf{x}_i$ and $\mathbf{x}_j$ in the form of

$$V_{ij} = \frac{k}{2}\left|(\mathbf{x}_i - \mathbf{x}_j) - (\mathbf{x}_{i0} - \mathbf{x}_{j0})\right|^2, \tag{2.1}$$

where k is the spring constant and $\mathbf{x}_{i0}$ is the equilibrium position of mass point i. This potential energy does not match the properties of fabric, but it is easy to analyze. We can show that a spring system with potential energy Eq. (2.1) has pure oscillatory motion and its eigen frequencies have an upper bound $\sqrt{2Mk/m}$, where M is the maximum number of neighbors a mass point can have. This upper bound of eigen frequencies plays an important role in the analysis of numerical stability and accuracy for schemes to solve the system.

As we mentioned in the introduction section, a fabric surface is considered as a membrane which is an idealized two dimensional manifold for which forces needed to bend it are negligible. Therefore at current stage we do not consider the bending energy. Model-I contains strong bending force and is not suitable for fabric modeling. For a realistic spring system to model the fabric surface, we have to assume that the spring force between two neighboring mass points is only proportional to the displacement from their equilibrium distance, the potential energy due to the relative displacement between two neighboring point mass $\mathbf{x}_i$ and $\mathbf{x}_j$ is given by

$$V_{ij} = \frac{k}{2}(|\mathbf{x}_i - \mathbf{x}_j| - l_{ij}^0)^2, \tag{2.2}$$

where $l_{ij}^0 = |\mathbf{x}_{i0} - \mathbf{x}_{j0}|$ is the equilibrium length between the two point masses. Here we have made a modification from the model used by Choi and Ko in that we allow both contraction and expansion forces while in Choi and Ko's equation, no force is applied if $|\mathbf{x}_i - \mathbf{x}_j| < l_{ij}^0$. Choi and Ko's choice does not conserve energy and would allow a surface shrink to a point without resistance. Such shrinking is unrealistic for a fabric surface. We call this system as Model-II.

The Lagrangians of the two systems are, therefore,

$$L = T - V = \sum_{i=1}^{N} \frac{1}{2} m |\dot{\mathbf{x}}_i|^2 - \frac{1}{4} \sum_{i=1}^{N} \sum_{j=1}^{N} k \big| (\mathbf{x}_i - \mathbf{x}_j) - (\mathbf{x}_{i0} - \mathbf{x}_{j0}) \big|^2 \eta_{ij} \tag{2.3}$$

for Model-I and

$$L = T - V = \sum_{i=1}^{N} \frac{1}{2} m |\dot{\mathbf{x}}_i|^2 - \frac{1}{4} \sum_{i=1}^{N} \sum_{j=1}^{N} k (|\mathbf{x}_i - \mathbf{x}_j| - l_{ij}^0)^2 \eta_{ij} \tag{2.4}$$

for Model-II, where η_{ij} is the adjacency coefficient between mass point i and j, and $\eta_{ij} = 1$ if mass points i and j are immediate neighbors, $\eta_{ij} = 0$ if $i = j$ or mass points i and j are not direct neighbors.

Apply Lagrangian equation

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) = \frac{\partial L}{\partial q_i}$$

to each mass point at $\mathbf{x}_i$, we have

$$m\frac{d\dot{\mathbf{x}}_i}{dt} = m\frac{d^2\mathbf{x}_i}{dt^2} = -\sum_{j=1}^{N} \eta_{ij} k \big((\mathbf{x}_i - \mathbf{x}_j) - (\mathbf{x}_{i0} - \mathbf{x}_{j0}) \big), \tag{2.5}$$

for Model-I and

$$m\frac{d\dot{\mathbf{x}}_i}{dt} = m\frac{d^2\mathbf{x}_i}{dt^2} = -\sum_{j=1}^{N} \eta_{ij} k \Big(\mathbf{x}_i - \mathbf{x}_j - l_{ij}^0 \mathbf{e}_{ij} \Big), \tag{2.6}$$

for Model-II, where $\mathbf{e}_{ij} = \frac{\mathbf{x}_i - \mathbf{x}_j}{|\mathbf{x}_i - \mathbf{x}_j|}$ is the unit vector from $\mathbf{x}_i$ to $\mathbf{x}_j$.

2.3 The forces in the two models

To see the difference between the two spring models, we rewrite the right hand side of Eq. (2.5) as the following

$$\begin{aligned} m\frac{d^2\mathbf{x}_i}{dt^2} &= -\sum_{j=1}^{N}\eta_{ij}k\Big(\mathbf{x}_i-\mathbf{x}_j-l_{ij}^0\mathbf{e}_{ij}+l_{ij}^0\mathbf{e}_{ij}-(\mathbf{x}_{i0}-\mathbf{x}_{j0})\Big) \\ &= -\sum_{j=1}^{N}\eta_{ij}k\Big((|\mathbf{x}_i-\mathbf{x}_j|-l_{ij}^0)\mathbf{e}_{ij}+l_{ij}^0(\mathbf{e}_{ij}-\mathbf{e}_{ij}^0)\Big). \end{aligned} \tag{2.7}$$

We may use $\mathbf{f}_i^s$ and $\mathbf{f}_i^r$ to represent the two forces in the right hand side of Eq. (2.7), that is,

$$\mathbf{f}_i^s = -\sum_{j=1}^{N}\eta_{ij}k(|\mathbf{x}_i-\mathbf{x}_j|-l_{ij}^0)\mathbf{e}_{ij}$$

and

$$\mathbf{f}_i^r = -\sum_{j=1}^{N}\eta_{ij}kl_{ij}^0(\mathbf{e}_{ij}-\mathbf{e}_{ij}^0).$$

It is easy to see that $\mathbf{f}_i^s$ is the restoring force due to stretching while $\mathbf{f}_i^r$ represents the restoring force due to bending. Therefore, if we introduce the third model, or Model-III:

$$m\frac{d^2\mathbf{x}_i}{dt^2} = \mathbf{f}_i^r, \tag{2.8}$$

and since Model-II is

$$m\frac{d^2\mathbf{x}_i}{dt^2} = \mathbf{f}_i^s, \tag{2.9}$$

we can then see that the force in Model-I is the superposition of the forces in Model-II and Model-III, that is, for Model-I

$$m\frac{d^2\mathbf{x}_i}{dt^2} = \mathbf{f}_i^s+\mathbf{f}_i^r. \tag{2.10}$$

Eq. (2.8) refers to bending only, Eq. (2.9) refers to stretching only (fabric), while Eq. (2.10) is the combined motion of stretching and bending. Therefore, the difference between Model-I and the fabric model (Model-II) is that the latter has no bending energy, thus no restoring force in the direction normal to the surface. We will show through numerical solution that for fabric surface, the tangential motion is oscillatory while the normal motion is not.

In the next section, we prove that the motion of Model-I is purely oscillatory and there exists an upper bound for the eigen frequencies of Model-I. The analysis for Model-II is more difficult. We will analyze the eigen frequencies only through the numerical simulations.

3 The oscillatory motion of spring systems

3.1 Eigen frequency of Model-I

We start by inspecting a system that is relatively easy to analyze. First, we consider the following system

$$m\frac{d^2\mathbf{x}_i}{dt^2}+\sum_{j=1}^{N}\eta_{ij}k\Big((\mathbf{x}_i-\mathbf{x}_j)-(\mathbf{x}_i^0-\mathbf{x}_j^0)\Big)=0,\quad i=1,2,\cdots,N, \tag{3.1}$$

where $\mathbf{x}_i^0$, $i=1,\cdots,N$ are the equilibrium positions of point mass i. This system can be simplified by the substitution $\mathbf{x}'_i=\mathbf{x}_i-\mathbf{x}_i^0$, $i=1,2,\cdots,N$, which yield

$$m\frac{d^2\mathbf{x}'_i}{dt^2}+\sum_{j=1}^{N}\eta_{ij}k\big((\mathbf{x}'_i-\mathbf{x}'_j)\big)=0,\quad i=1,2,\cdots,N. \tag{3.2}$$

For simplicity, we will omit the prime in Eq. (3.2). Here we will use:

- $\mathbf{x}_i=(x_i,y_i,z_i)$, $i=1,\cdots,N$ — the coordinates of the mass points;
- k — the spring constant;
- m — the mass of a single point;
- $H=\{\eta_{ij}\}_{N\times N}$ — the adjacency matrix.

It is obvious that the adjacency matrix H is symmetric, that is, $\eta_{ij}=\eta_{ji}$. In this model we assume that no points will collide. Then we can set up the equations which govern the system

$$\begin{aligned}&\frac{\mathrm{d}\mathbf{x}_i}{\mathrm{d}t}=\mathbf{v}_i,\\&\frac{\mathrm{d}\mathbf{v}_i}{\mathrm{d}t}=-\frac{k}{m}\sum_{j\neq i}\eta_{ij}(\mathbf{x}_i-\mathbf{x}_j)=-\sum_{j\neq i}a_{ij}(\mathbf{x}_i-\mathbf{x}_j),\end{aligned}$$

where $\mathbf{v}_i=(u_i,v_i,w_i)$ is the velocity of the mass point i. Here we can see $a_{ij}=a_{ji}\geq 0$. In matrix form we have

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix}\mathbf{x}_1\\\vdots\\\mathbf{x}_N\\\mathbf{v}_1\\\vdots\\\mathbf{v}_N\end{pmatrix}=\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix}\mathbf{x}\\\mathbf{v}\end{pmatrix}=\begin{pmatrix}\mathbf{0}_{3N\times 3N}&\mathbf{I}_{3N}\\\mathbf{R}&\mathbf{0}_{3N\times 3N}\end{pmatrix}\begin{pmatrix}\mathbf{x}\\\mathbf{v}\end{pmatrix},$$

or

$$\frac{d\mathbf{u}}{dt}=A\mathbf{u}, \tag{3.3}$$

where

$$\mathbf{u}=\begin{pmatrix}\mathbf{x}\\ \mathbf{v}\end{pmatrix},\quad \mathbf{A}=\begin{pmatrix}\mathbf{0}_{3N\times 3N} & \mathbf{I}_{3N}\\ \mathbf{R} & \mathbf{0}_{3N\times 3N}\end{pmatrix}$$

and

$$\mathbf{R}=\begin{pmatrix} -\sum_{j\neq 1}a_{1j}\mathbf{I}_3 & a_{12}\mathbf{I}_3 & \cdots & a_{1N}\mathbf{I}_3\\ a_{21}\mathbf{I}_3 & -\sum_{j\neq 2}a_{2j}\mathbf{I}_3 & \cdots & a_{3N}\mathbf{I}_3\\ \vdots & \vdots & \ddots & \vdots\\ a_{N1}\mathbf{I}_3 & a_{N2}\mathbf{I}_3 & \cdots & -\sum_{j\neq N}a_{Nj}\mathbf{I}_3 \end{pmatrix}.$$

We can show that all the eigenvalues of $\mathbf{A}$ are either zero or pure imaginary. The eigenvalues of $\mathbf{A}$ are the roots of equation $\det(\mathbf{A}-\lambda\mathbf{I}_{6N})=0$

$$\begin{aligned}
&\det(\mathbf{A}-\lambda\mathbf{I}_{6N})\\
=&\begin{vmatrix}-\lambda\mathbf{I}_{3N} & \mathbf{I}_{3N}\\ \mathbf{R} & -\lambda\mathbf{I}_{3N}\end{vmatrix}=\begin{vmatrix}-\lambda\mathbf{I}_{3N} & \mathbf{I}_{3N}\\ \mathbf{R}-\lambda^2\mathbf{I}_{3N} & \mathbf{0}_{3N\times 3N}\end{vmatrix}\\
=&(-1)^{3N}\begin{vmatrix}\mathbf{I}_{3N} & -\lambda\mathbf{I}_{3N}\\ \mathbf{0}_{3N\times 3N} & \mathbf{R}-\lambda^2\mathbf{I}_{3N}\end{vmatrix}=(-1)^{3N}\left|\mathbf{R}-\lambda^2\mathbf{I}_{3N}\right|\\
=&(-1)^{6N}\left|\lambda^2\mathbf{I}_{3N}-\mathbf{R}\right|\\
=&\begin{vmatrix}\Big(\lambda^2+\sum\limits_{j\neq 1}a_{1j}\Big)\mathbf{I}_3 & -a_{12}\mathbf{I}_3 & \cdots & -a_{1N}\mathbf{I}_3\\ -a_{21}\mathbf{I}_3 & \Big(\lambda^2+\sum\limits_{j\neq 2}a_{2j}\Big)\mathbf{I}_3 & \cdots & -a_{3N}\mathbf{I}_3\\ \vdots & \vdots & \ddots & \vdots\\ -a_{N1}\mathbf{I}_3 & -a_{N2}\mathbf{I}_3 & \cdots & \Big(\lambda^2+\sum\limits_{j\neq N}a_{Nj}\Big)\mathbf{I}_3\end{vmatrix}\\
=&\begin{vmatrix}\lambda^2+\sum\limits_{j\neq 1}a_{1j} & -a_{12} & \cdots & -a_{1N}\\ -a_{21} & \lambda^2+\sum\limits_{j\neq 2}a_{2j} & \cdots & -a_{2N}\\ \vdots & \vdots & \ddots & \vdots\\ -a_{N1} & -a_{N2} & \cdots & \lambda^2+\sum\limits_{j\neq N}a_{Nj}\end{vmatrix}^3=0,
\end{aligned}$$

or equivalently

$$\begin{vmatrix}\lambda^2+\sum\limits_{j\neq 1}a_{1j} & -a_{12} & \cdots & -a_{1N}\\ -a_{21} & \lambda^2+\sum\limits_{j\neq 2}a_{2j} & \cdots & -a_{2N}\\ \vdots & \vdots & \ddots & \vdots\\ -a_{N1} & -a_{N2} & \cdots & \lambda^2+\sum\limits_{j\neq N}a_{Nj}\end{vmatrix}=0. \tag{3.4}$$

Before we reach the conclusion we need to review the Levy-Desplanques theorem.

3.2 Levy-Desplanques theorem

If a matrix is strictly diagonally dominant, then it is nonsingular.

Then we finish the proof of our claim. Put

$$\mathbf{B}=\begin{pmatrix} \sum_{j\neq 1}a_{1j} & -a_{12} & \cdots & -a_{1N} \\ -a_{21} & \sum_{j\neq 2}a_{2j} & \cdots & -a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ -a_{N1} & -a_{N2} & \cdots & \sum_{j\neq N}a_{Nj} \end{pmatrix}.$$

First we can see that for any eigenvalue λ of **A**, it corresponds to the eigenvalue μ of **B** by $\lambda=\sqrt{-\mu}$. Now what we want to prove that all the eigenvalues of **B** are 0 or positive real numbers.

Since **B** is symmetric, all the eigenvalues of **B** are real numbers.

If a real number $\mu<0$ is an eigenvalue of **B**, it must satisfy $\det(\mathbf{B}-\mu\mathbf{I}_N)=0$. However, considering the fact that $a_{ij}\geq 0$, we can easily conclude that the matrix

$$\begin{pmatrix} \sum_{j\neq 1}a_{1j}-\mu & -a_{12} & \cdots & -a_{1N} \\ -a_{21} & \sum_{j\neq 2}a_{2j}-\mu & \cdots & -a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ -a_{N1} & -a_{N2} & \cdots & \sum_{j\neq N}a_{Nj}-\mu \end{pmatrix}$$

is strictly diagonally dominant. According to Levy-Desplanques Theorem, the matrix is nonsingular, or

$$\det(\mathbf{B}-\mu\mathbf{I}_N)\neq 0.$$

Contradiction! That means, all eigenvalues of **B** are real numbers and they are 0 or positive. Since $\lambda=\sqrt{-\mu}$, we come to the conclusion that all the eigenvalues of **A** are either 0 or pure imaginary, and the system is neutrally stable. We will write $\lambda=i\omega$, where $\omega=\sqrt{\mu}$.

3.3 Bound of eigen values of Model-I

Assume that every mass point has at most M neighbors, and that the displacement of all points do not vary too much, we can estimate the upper bound of the eigenvalues by estimating that of matrix B. According to Gershgorin circle theorem, all the eigenvalues of B lie within the circles D_i, $i=1,\cdots,N$ where

$$D_i=\left\{\mu;\left|\mu-\Big(-\sum_{j\neq i}a_{ij}\Big)\right|\leq\sum_{j\neq i}a_{ij}\right\}.$$

Then we can see

$$|\mu|\leq 2\left|\sum_{j\neq i}a_{ij}\right|\approx 2Ma=\frac{2Mk}{m}. \tag{3.5}$$

In other words, all the eigenvalues of A satisfy

$$|\lambda|=\sqrt{|\mu|}\leq\sqrt{\frac{2Mk}{m}}. \tag{3.6}$$

The numerical experiments show that this upper bound is actually a little too high. The numerical solutions suggest that the minimum upper bound should be

$$\omega=|\lambda|=\sqrt{|\mu|}\leq\sqrt{\frac{Mk}{m}}. \tag{3.7}$$

3.4 Oscillatory motion of Model-II

To analyze Model-II, we rewrite the Eq. (2.6) as the following

$$m\frac{d\dot{\mathbf{x}}_i}{dt}=-\sum_{j=1}^{N}\eta_{ij}k\left(\mathbf{x}_i-\mathbf{x}_j-l_{ij}^0\mathbf{e}_{ij}\right)=-\sum_{j=1}^{N}\eta_{ij}k\frac{\Delta l_{ij}}{l_{ij}}\left(\mathbf{x}_i-\mathbf{x}_j\right), \tag{3.8}$$

where $l_{ij}=|\mathbf{x}_i-\mathbf{x}_j|$ and $\Delta l_{ij}=l_{ij}-l_{ij}^0$. Since Δl_{ij} can be either positive or negative, we cannot guarantee that all the eigen values of the system are imaginary.

We write Eq. (2.6) as

$$m\frac{d^2\mathbf{x}_i}{dt^2}=-\sum_{j\neq i}a_{ij}(l_{ij})(\mathbf{x}_i-\mathbf{x}_j), \tag{3.9}$$

where

$$a_{ij}=\eta_{ij}k\left(1-\frac{l_{ij}^0}{l_{ij}}\right).$$

We linearize Eq. (3.9) and obtain

$$m\frac{d^2\delta\mathbf{x}_i}{dt^2}=-\sum_{j\neq i}\left\{\delta a_{ij}(l_{ij})(\mathbf{x}_i-\mathbf{x}_j)+a_{ij}(\delta\mathbf{x}_i-\delta\mathbf{x}_j)\right\}. \tag{3.10}$$

Using

$$\delta a_{ij}(l_{ij})=a'(l_{ij})\delta l_{ij}=\eta_{ij}k\frac{l_{ij}^0}{l_{ij}^2}\delta l_{ij}$$

and

$$\begin{aligned}\delta l_{ij}&=\delta\left(\sum_d(x_i^d-x_j^d)^2\right)^{1/2}\\&=\frac{\sum_d(x_i^d-x_j^d)(\delta x_i^d-\delta x_j^d)}{\left(\sum_d(x_i^d-x_j^d)^2\right)^{1/2}}=\frac{1}{l_{ij}}(\mathbf{x}_i-\mathbf{x}_j)\cdot(\delta\mathbf{x}_i-\delta\mathbf{x}_j),\end{aligned}$$

we obtain

$$\begin{aligned}\delta a_{ij}(\mathbf{x}_i-\mathbf{x}_j)&=\eta_{ij}k\left[\frac{l_{ij}^0}{l_{ij}^3}(\mathbf{x}_i-\mathbf{x}_j)\big((\mathbf{x}_i-\mathbf{x}_j)\cdot(\delta\mathbf{x}_i-\delta\mathbf{x}_j)\big)\right]\\&=\eta_{ij}k\left[\frac{l_{ij}^0}{l_{ij}}\mathbf{e}_{ij}\big(\mathbf{e}_{ij}\cdot(\delta\mathbf{x}_i-\delta\mathbf{x}_j)\big)\right]\\&=\eta_{ij}k\left[\frac{l_{ij}^0}{l_{ij}}\mathbf{e}_{ij}\otimes\mathbf{e}_{ij}(\delta\mathbf{x}_i-\delta\mathbf{x}_j)\right].\end{aligned}\tag{3.11}$$

Substituting Eq. (3.11) into Eq. (3.10), we have

$$m\frac{d^2\delta\mathbf{x}_i}{dt^2}=-\sum_{j\neq i}\eta_{ij}k\left(\frac{l_{ij}^0}{l_{ij}}\mathbf{e}_{ij}\otimes\mathbf{e}_{ij}+\left(1-\frac{l_{ij}^0}{l_{ij}}\right)I\right)(\delta\mathbf{x}_i-\delta\mathbf{x}_j).\tag{3.12}$$

The summation on right hand side of Eq. (3.12) represents the superposition of forces due to each neighboring mass point. Using $\mathbf{e}_{ij}$ to make dot product for each term, we obtain the tangential component of acceleration

$$\begin{aligned}f_{ij}^t&=-\mathbf{e}_{ij}\cdot\eta_{ij}k\left(\frac{l_{ij}^0}{l_{ij}}\mathbf{e}_{ij}\otimes\mathbf{e}_{ij}+\left(1-\frac{l_{ij}^0}{l_{ij}}\right)I\right)(\delta\mathbf{x}_i-\delta\mathbf{x}_j)\\&=-\eta_{ij}k\big(\mathbf{e}_{ij}\cdot(\delta\mathbf{x}_i-\delta\mathbf{x}_j)\big).\end{aligned}\tag{3.13}$$

Eq. (3.13) indicates that the force on a mass point along each direction to its neighbors is restoring, therefore the motion is oscillatory.

3.5 Coupling with external drive

The interaction between the external driving force and the mass points of the fabric surface deserves careful study in computation. The system described by Eq. (2.5) conserves the total energy. However in dynamic simulation of the fabric surface, the total energy may increase and the system can be excited due to stretching and compression by external forces. The external force not only adds to the acceleration of the macro-scale motion of the fabric surface, it also displaces the mass points in the tangential directions and incites internal energy for the spring system. The restoring force between each pair of neighboring mass points of the spring system serves as the self-adjustment to satisfy the fabric constraint and Young's modulus. However, when the internal energy of the spring system is too high, the system may be dominated by the random meso-scale motion of the mass points. Therefore adding a damping force will help to stabilize the system.

When there is an external velocity field $\mathbf{v}^e$, we define the external impulse as $\mathbf{I}_i^e=m\mathbf{v}_i^e$, where $\mathbf{v}_i^e$ is the external driving velocity at point $\mathbf{x}_i$. At any given time, we can solve the equations of the spring system and obtain the internal impulse $\mathbf{I}_i^s$. Since the spring force is a function of the relative position of the point mass with respect to its neighbors, we can use the superposition principle and add to the total impulse

$$\mathbf{I}_i = \mathbf{I}_i^e + \mathbf{I}_i^s. \tag{3.14}$$

Our method is to apply damping to the internal impulse only.

In a fluid interaction, the fabric surface points may be driven by three forces, the gravitational force due to the weight of the fabric surface point masses, the pressure force from the surrounding fluid, and the internal force, or the spring restoring force and friction (to prevent the spring system becoming over excited). We separate the impulse on a mass point into three components, the gravitational impulse, the pressure force impulse and the internal impulse due to neighboring point mass in the spring system, that is

$$\mathbf{I}_i = \mathbf{I}_i^g + \mathbf{I}_i^p + \mathbf{I}_i^s. \tag{3.15}$$

The interaction between the fabric surface and fluid is through the normal component of the force on each mass point. At every time step, the fluid exerts an impulse to the mass points, but this part of the impulse is balanced by the gravitational impulse and the restoring force of the spring. The normal component of the superposition of the three forces feeds back to the fluid in the following step. The result is that the momentum exchange between the fabric surface and the fluid is equal in magnitude and opposite in directions, a requirement by Newton's third law.

To prevent the spring system getting into over-excited state, we add a damping force to the system. Therefore, the complete system of equations is the following

$$\begin{aligned}
\frac{d\mathbf{v}_i}{dt} &= -\frac{1}{m}\sum_{j=1}^{N} \eta_{ij} k \left(|\mathbf{x}_i - \mathbf{x}_j| - l_{ij}^0 \right) \mathbf{e}_{ij} + \mathbf{f}_i^e - \kappa \mathbf{v}_i^s, \\
\frac{d\mathbf{x}_i}{dt} &= \mathbf{v}_i,
\end{aligned}$$

where $\mathbf{f}_i^e$ is the external force, κ is the damping coefficient and $\mathbf{v}_i^s$ is the velocity component due to the spring impulse $\mathbf{v}_i^s = \mathbf{I}_i^s / m$.

3.6 Collision handling

Collision handling is a challenging work to many researchers in fabric modeling. Recently many authors have presented their algorithms and techniques on this tedious but important work. Bez *et al*. [7] introduced the topological mapping approach to collision detection for the drape simulation work. Bridson *et al*. [11, 12] presented an algorithm to efficiently and robustly process collisions, contact and friction in cloth simulation. Bargmann [6] presented a new approach to collision detection and collision response of cloth onto deformable volumes, along with a self-collision algorithm to handle collisions of the cloth with itself.

Our *FronTier* library has a set of functions for detection of collisions, although the existing functions were designed to resolve collisions through topological bifurcation or merging using the locally grid based method [16]. For the modeling of fabric surface, we modify the functions to reflecting the colliding marker points in the normal direction of the surface. The detection uses an efficient hashing method and is of order $\mathcal{O}(nN)$, where n is the number of triangles hashed by one grid block and N is the total number of triangles.

3.7 Numerical stability of the spring system

The spring system is an energy conservative ODE system if we set the damping coefficient equal to zero. The numerical methods for solving system of ordinary differential equations is now a curriculum topic in numerical analysis courses. Although textbooks such as the one by Butcher [13] discussed method for the oscillatory linear system, few have mentioned about the crucial role of the upper bound of the eigen frequency in the stability and accuracy analysis for the conservative ODE system. In this section, we give a detailed derivation of the truncation error of the Runge-Kutta methods and their dependency on the frequency of the oscillation.

In one dimension or for Model-I (Eq. (2.5)), the equations of spring system can be transformed into homogeneous equation if we redefine x_i as the relative displacement from its equilibrium position. Therefore, we will first consider the numerical stability of Eq. (3.3)

$$\frac{d\mathbf{u}}{dt}=A\mathbf{u},$$

here we have used $\mathbf{u}=\{\mathbf{x}_1^T,\mathbf{x}_2^T,\cdots,\mathbf{x}_N^T,\mathbf{v}_1^T,\mathbf{v}_2^T,\cdots,\mathbf{v}_N^T\}^T$. If we can diagonalize the matrix $A=T^{-1}\Lambda T$, and introduce $\mathbf{w}=T\mathbf{u}$, we can obtain the decoupled equation

$$\frac{d\mathbf{w}}{dt}=\Lambda\mathbf{w}. \tag{3.16}$$

Therefore to consider the stability of the system Eq. (3.16), we should consider the scalar equation

$$\frac{dy}{dt}=\lambda y. \tag{3.17}$$

The Euler method for Eq. (3.17)

$$y^{n+1}=y^n(1+\lambda\Delta t) \tag{3.18}$$

and its stability is well known if λ is real. However, for a conservative system like the spring equations with damping coefficient equal to zero, all the eigen values of A are imaginary. Therefore, we will need to consider the following equation

$$\frac{dy}{dt}=i\omega y, \tag{3.19}$$

with $\omega=|\lambda|=\sqrt{\mu}$. If we use the Euler forward method for Eq. (3.19), that is

$$y^{n+1}=y^n(1+i\omega\Delta t), \tag{3.20}$$

since the square mode of the recursive factor $\chi^2=|1+i\omega\Delta t|^2=1+\omega^2\Delta t^2>1$, therefore the solution will be amplified after each time step by χ. To reduce this unphysical amplification, we need to make $\omega\Delta t\ll 1$ by reducing Δt.

With same Δt, an increasing order of the scheme will reduce this amplification factor effectively. For the second order predictor-corrector method

$$y^*=y^n(1+i\omega\Delta t), \tag{3.21a}$$

$$y^{n+1}=y^n+\frac{1}{2}i\omega\Delta t(y^n+y^*)=\Big(1-\frac{1}{2}\omega^2\Delta t^2+i\omega\Delta t\Big)y^n \tag{3.21b}$$

will have $\chi^2=1+\frac{1}{4}(\omega\Delta t)^4$. In particular, for the fourth order Runge-Kutta method, that is

$$\begin{aligned}
&y^{n+1}=y^n+\frac{\Delta t}{6}(K_1+2K_2+2K_3+K_4),\\
&K_1=i\omega y^n,\\
&K_2=i\omega\Big(y^n+\frac{\Delta t}{2}K_1\Big)=i\omega y^n\Big(1+\frac{i\omega\Delta t}{2}\Big),\\
&K_3=i\omega\Big(y^n+\frac{\Delta t}{2}K_2\Big)=i\omega y^n\Big(1+\frac{i\omega\Delta t}{2}+\frac{(i\omega\Delta t)^2}{4}\Big),\\
&K_4=i\omega(y^n+\Delta tK_3)=i\omega y^n\Big(1+i\omega\Delta t+\frac{(i\omega\Delta t)^2}{2}+\frac{(i\omega\Delta t)^3}{4}\Big),
\end{aligned}$$

and we can see

$$\begin{aligned}
y^{n+1}&=y^n+\frac{h}{6}(K_1+2K_2+2K_3+K_4)\\
&=\Big(1+i\omega\Delta t+\frac{(i\omega\Delta t)^2}{2}+\frac{(i\omega\Delta t)^3}{6}+\frac{(i\omega\Delta t)^4}{24}\Big)y^n\\
&=\Big(1+i\omega\Delta t-\frac{(\omega\Delta t)^2}{2}-\frac{i(\omega\Delta t)^3}{6}+\frac{(\omega\Delta t)^4}{24}\Big)y^n\\
&=\Big(\Big(1-\frac{(\omega\Delta t)^2}{2}+\frac{(\omega\Delta t)^4}{24}\Big)+i\Big(\omega\Delta t-\frac{(\omega\Delta t)^3}{6}\Big)\Big)y^n.
\end{aligned}$$

The growth factor is

$$\begin{aligned}
\chi^2&=\Big|\Big(1-\frac{(\omega\Delta t)^2}{2}+\frac{(\omega\Delta t)^4}{24}\Big)+i\Big(\omega\Delta t-\frac{(\omega\Delta t)^3}{6}\Big)\Big|^2\\
&=\Big(1-\frac{(\omega\Delta t)^2}{2}+\frac{(\omega\Delta t)^4}{24}\Big)^2+\Big(\omega\Delta t-\frac{(\omega\Delta t)^3}{6}\Big)^2\\
&=1-\frac{1}{72}(\omega\Delta t)^6+\frac{1}{576}(\omega\Delta t)^8.
\end{aligned} \tag{3.22}$$

The above calculation shows that the fourth order Runge-Kutta method for the oscillatory system is not only stable, but also very accurate.

Model-I is not a suitable model for fabric because it includes too much bending energy and the restoring force normal to the surface. Since Model-II is a nonlinear system, proof similar to the one for Model-I is difficult. However, Eq. (3.13) shows that for Model-II, the force tangential to the fabric surface follows the homogeneous system Eq. (3.2), therefore we perform a set of numerical tests and the results show that for the fabric system (Model-II), the motion of mass points in the tangential direction is oscillatory whose eigen frequency is also bounded. In the direction normal to the surface, the motion is not oscillatory in general and its rate of change is much smaller than that in the tangential directions. Therefore as long as the scheme is stable in the tangential direction, it is automatically stable for the motion in the normal direction.

4 Numerical results

4.1 Young's modulus

The stiffness coefficient of the spring k is related to the Young's modulus. which is defined as

$$E=\frac{\sigma}{\epsilon}=\frac{F}{A}\frac{l}{\Delta l}, \tag{4.1}$$

where σ and ϵ are the stress and strain on the elastic material, A is the cross sectional area of the elastic body, F is the force applied to the body perpendicular to the area A, l and Δl are the equilibrium and stretched length of the body. For the fabric surface, in order to compare the Young's modulus to the spring constant in our model, we introduce the Young's surface modulus. This is to use Young's modulus to multiply the thickness of the fabric surface. That is

$$E_s=\frac{F}{D}\frac{l}{\Delta l}, \tag{4.2}$$

where D is the cross sectional length of a rectangular surface. The Young's surface modulus E_s (multiplying a unit length) should equal to the spring constant. However we have carried out the numerical simulation by applying a weight uniformly at the end of a plane fabric surface. The resulting shape of the fabric is concave inward in the middle section Fig. 2. The computed Young's surface modulus E_s (times a unit length) is about 1.1 times of the spring constant. The detailed analysis is more complicated, therefore we have used $E_s\approx 1.1k_s$ as the empirical value in our simulations.

4.2 The oscillatory motion of fabric spring model

In Section 3, we analyzed the oscillatory motion of Model-I and proved that for such system, the eigen values of its coefficient matrix are pure imaginary and these eigen values

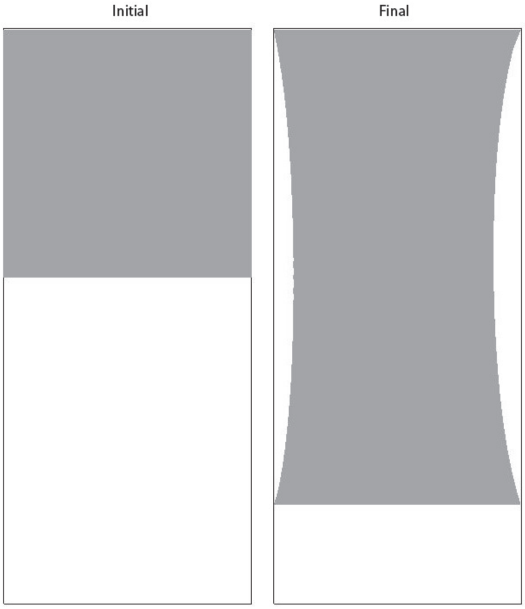

Figure 2: Numerical simulation of Young's modulus on surface. A force is uniformly applied to the bottom boundary and the surface is pulled downward. The middle part of the surface is deformed inward resulting in a concavity of the two vertical sides. The resulting Young's surface modulus E_s is about 1.1 times of the spring constant k_s. In this simulation, we set $k/m = 10^5$ and the friction constant $\kappa = 0.1$.

are bounded by Eq. (3.5). The fabric model (Model-II) is nonlinear and it is very difficult to prove the same proposition for Model-I. However, Eq. (3.13) indicates that in the tangential directions, the force on mass point in Model-II is the same as for Model-I, therefore we project that in Model-II, the motion of the mass points is identical to that in Model-I, while the motion of mass points in the normal direction is different.

To demonstrate this conjecture, we carried out the numerical simulation with a circular fabric surface whose boundary is fixed. We perturb the surface to let it start a drum-like motion (Fig. 3). We take a sample point from the surface and recorded its relative coordinates in horizontal directions (x and y) and the vertical direction (z). Fig. 4 shows the recorded motion in three directions of the coordinate system. This figure shows that the motions along the x and y directions (almost tangential to the fabric surface) are highly

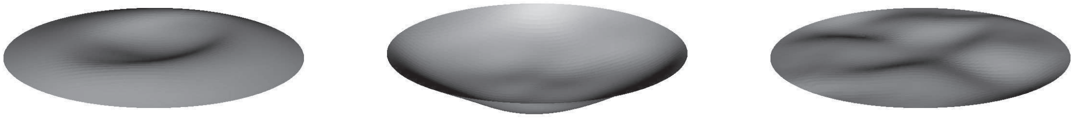

Figure 3: The plots in this figure show a fabric surface vibrating motion. The surface is fixed at the circular boundary. An initial perturbation is applied to the fabric surface. We use this simulation to study the motion of mass point tangential to the surface and normal to the surface. The spring constant in this case is $k = 1000$ and point mass $m = 0.625$.

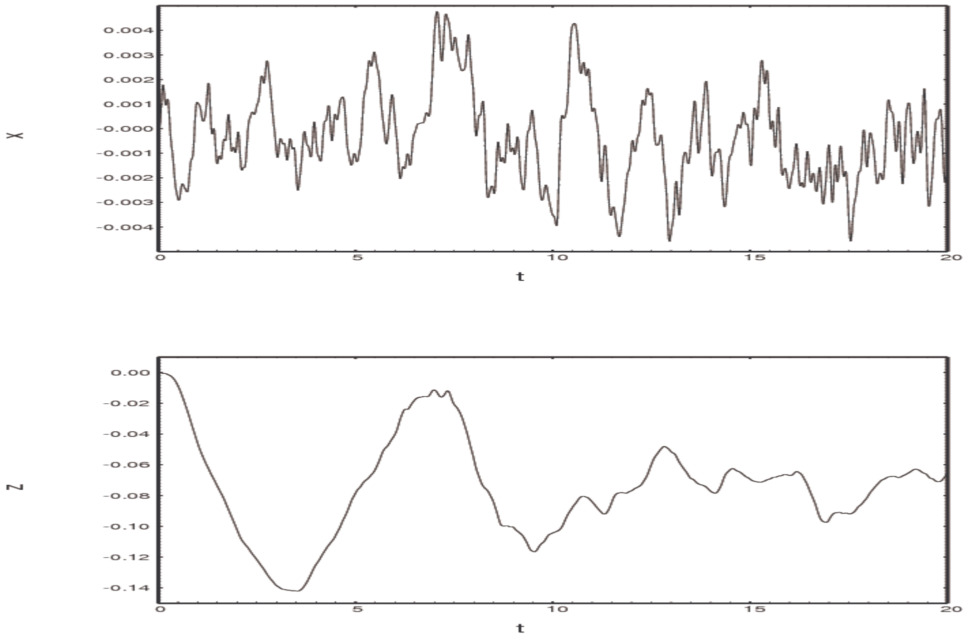

Figure 4: Relative displacement of a sample point on the fabric surface as a function of time. The top is the displacement almost tangential to the surface (the x-coordinate), the bottom is the displacement almost normal to the fabric surface. The tangential motion is oscillatory whose frequency is bounded by $\sqrt{Mk/m}$, with M approximately equal to 7. The normal motion is not oscillatory in general. There is no restoring force in the normal direction for fabric surface. The spring constant and point mass are the same as in Fig. 3.

oscillatory, while the motion along the z direction (almost normal to the fabric surface) has much smaller frequencies. In general, the motion along the z direction does not have to be oscillatory.

We analyzed the spectra of the oscillatory motion in the tangential direction of the fabric surface and found that the frequencies of the oscillatory modes are indeed bounded. Fig. 5 shows the spectra of the tangential oscillatory motion. The three plots in Fig. 5 are for runs with $k=1000$, and $m=10,2.5,0.625$ respectively. The three vertical lines represent the cut-off frequency in each run. They are at $\nu_c=\omega_c/2\pi=\sqrt{Mk/m}/2\pi=4.21,8.42,16.84$ respectively. Here we have used $M=7$, while the average number of neighbors of each mass point is 6. Our simulations show that the upper bound of the frequency is only dependent on the ratio of k/m and independent of the total number of mass points in the system.

Fig. 5 proved that there is indeed an upper bound for the eigen frequency of the oscillatory motion in the spring system. However, this upper bound appears to be only $1/\sqrt{2}$ of the value given by Eq. (3.7), which we derived from the Gershgorin circle theorem. In another word, the bound given by Eq. (3.5) is not optimal. A better proof may be needed to show the minimum upper bound given by the numerical solutions.

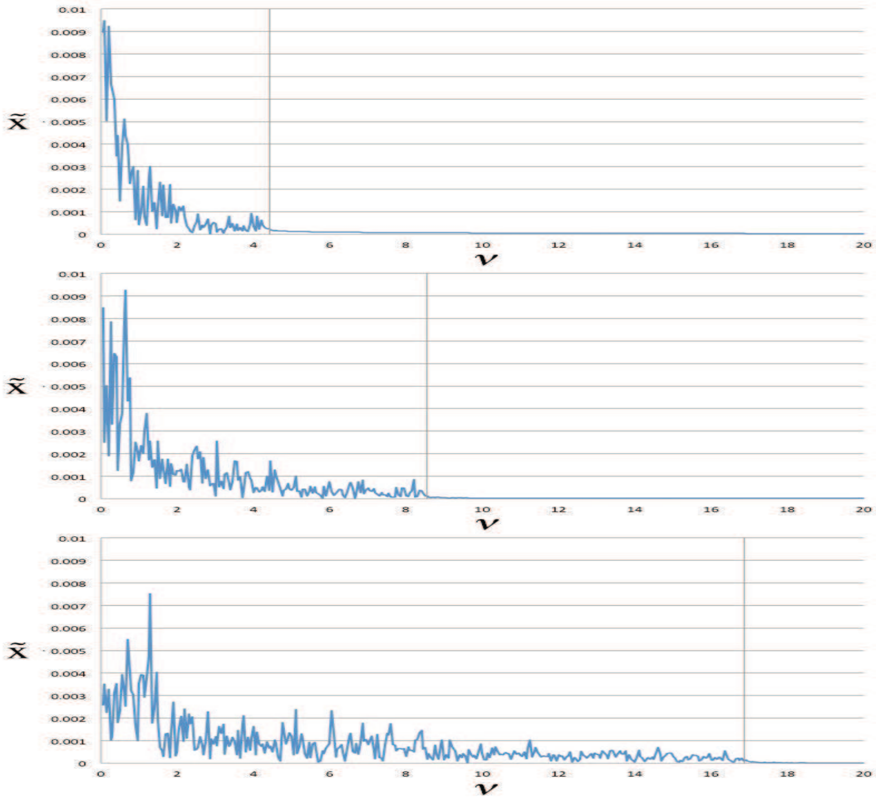

Figure 5: The three plots in the figure is the Fourier transformation of the tangential displacement function. The vertical line in each plot is the cut-off frequency $\omega_c = \sqrt{Mk/m}$ with $M=7$. The spring constant in all three cases is 1000, and the point masses for three cases from top to bottom are 10,2.5,0.625 respectively. These plots prove that the tangential oscillation frequency is bounded by $\omega_c = \sqrt{Mk/m}$.

4.3 First, second and fourth order schemes

From both the analysis in Section 3.7 and the numerical comparison, we find that there is a dramatic different between using the first order scheme and the fourth order scheme, the latter is only four times more expensive than the former.

Fig. 6 shows the comparison between second order and fourth order schemes on Eq. (3.19). The simulation shows that with $\mu\Delta t = 0.1$, the 4th order method gives sufficiently conserved solution even after 40,000 time steps while the second order method amplifies the motion and breaks energy conservation very quickly. A reduction of time step does not help the second scheme. The first order scheme is much worse due to Eq. (3.20). Our conclusion is that the use of fourth order scheme is not only stable but also very accurate. The comparison among first, second and fourth order methods over 400, 4000, and 40000 time steps are summarized by Table 1.

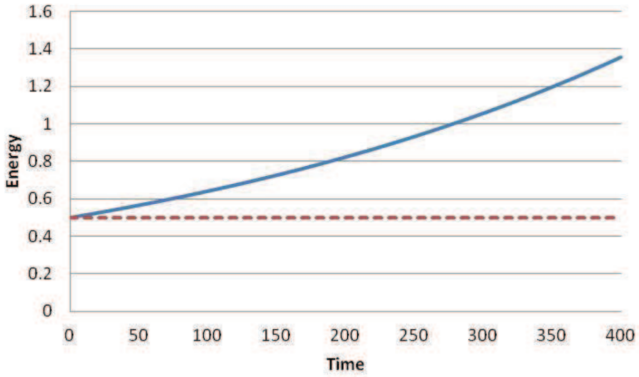

Figure 6: Comparison of the second and fourth order schemes for Eq. (3.19) with $\omega\Delta t = 0.1$ over 40,000 time steps. The plot shows the comparison of the total energy. The numerical results show that the second order predictor-corrector method added 1.7 time of the total energy to the initial value after 40,000 time steps while the fourth order Runge-Kutta method only changed 0.05 percent.

Table 1: Comparison of numerical schemes for ODE Eq. (3.19) with $\omega\Delta t = 0.1$ over 400, 4000, and 40000 steps respectively. The values in the table are for $\frac{\Delta E}{E_0}$, where E_0 is the total energy at $t=0$, and $\Delta E = E(T) - E_0$ is the difference between the final total energy the initial total energy.

Order of Scheme	N=400	N=4000	N=40000
First Order	52.5	9.6×10^{16}	∞
Second Order	0.01	0.105	1.7
Fourth Order	-6.0×10^{-6}	-5.6×10^{-5}	-5.5×10^{-4}

4.4 Wrinkling, draping and parachute inflation

To demonstrate how the spring mass system reacts to the stretching and wrinkling when a fabric surface is driven by an external force, we presented several simulation examples.

The first example is a square fabric surface driven by a hyperbolic velocity field. Fig. 7 is the top view of the wrinkling of the edges when the velocity at the center is larger. The spring model does not conserve the total area exactly, but the restoring force preserves the total area approximately. The second example is the simulation of a square table cloth draping from an oval table. Under the gravitational force, the fabric constraint automatically adjusts the parts of the cloth. Collision is handled by functions in the front tracking library. Fig. 8 shows the simulation of a table cloth of draping under the gravitational field.

The main motivation of studying the spring mass model for fabric surface is to study the parachute dynamics for the air delivery system. In this application, the fabric surface is an immersed surface in the incompressible fluid solver. We couple the PDE solver for the Navier-Stokes equation with the ODE solver for the point mass system through the impulse method. The physical interaction between the parachute canopy and the fluid is through the exchange of the normal impulse of the mass points in the fabric surface.

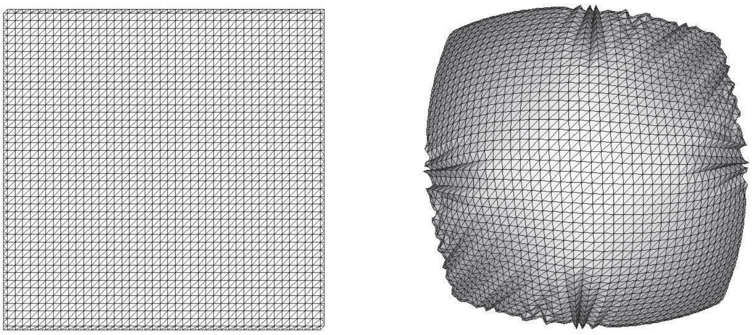

Figure 7: Top view of the evolution of a square cloth driven by a parabolic velocity field $v=v_0(x^2+y^2)$. While the cloth responds to the external velocity field, the spring system automatically adjusts the total internal energy toward minimum state resulting in the wrinkles at the edges of the cloth. The characteristic eigen frequency for the fabric model is $\sqrt{k/m}=100$ and the friction constant is zero.

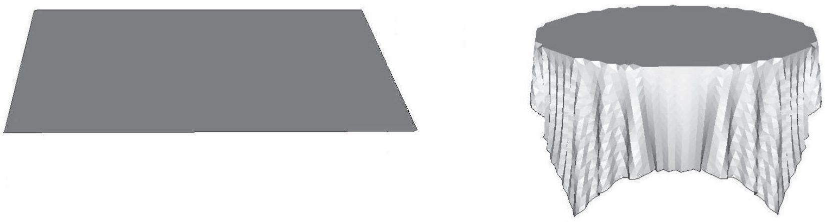

Figure 8: Simulation of a table cloth draping under the gravitational force. The fabric constraint automatically adjusts the parts of the cloth. The spring model of the fabric gives a realistic motion of the cloth. The characteristic eigen frequency for the fabric model in this simulation is $\sqrt{k/m}=1000$ and the friction constant is $\kappa=0.1$.

Fig. 9 shows the inflation of the G-11 cargo parachute and Fig. 10 shows the inflation of the cross parachute. The validation study of the parachute system is presented in a different paper [21].

5 Conclusions

We use the front tracking data structure and functionalities to model the dynamic motion of fabric material. Our objective is to use this model for the computational study of the air delivery system such as the parachute system. We established the computational platform by using the spring mass system. We considered two spring systems, the linear system and the fabric system, the latter has no bending energy and is a suitable model for fabric material. For the linear system (Model-I), we have proved, through the Levy-Desplanques Theorem and the Gershgorin circle theorem, that all the eigen values of the coefficient matrix are imaginary and therefore the motion is pure oscillatory, and there

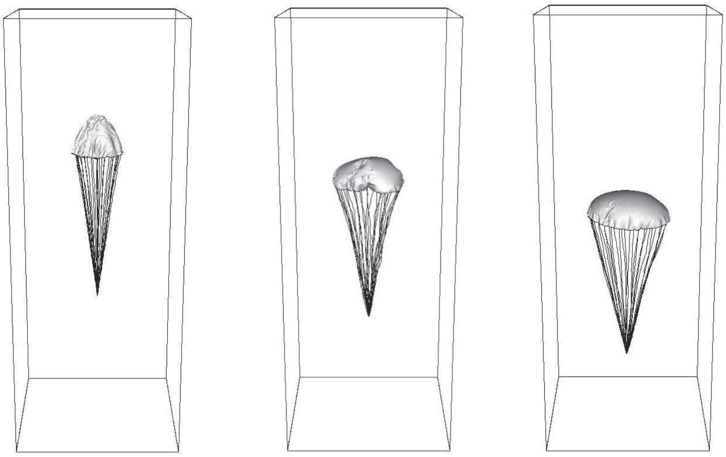

Figure 9: Spring mass model of fabric surface applied to the simulation of G-11 parachute inflation process. The three plots show the G-11 parachute opening from a half-folded state. Both the canopy and the string chords are modeled by the spring mass system. This simulation uses spring constant $k=5000$ which matches the Young's modulus of parachute canopy. The spring point mass is $m=0.1$ and the damping coefficient is $\kappa=0.001$ for the canopy and $\kappa=0.01$ for the string chords.

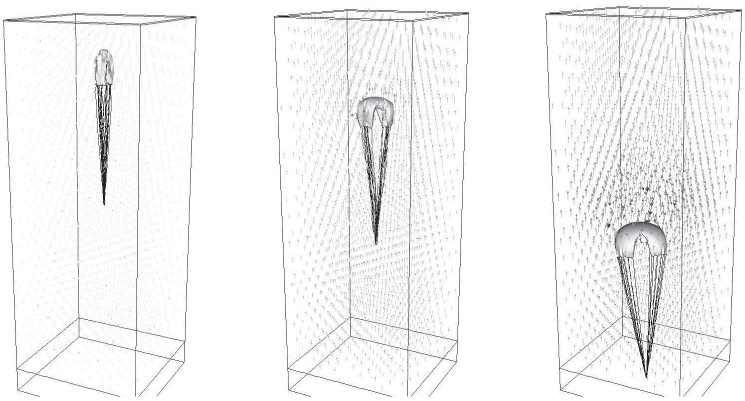

Figure 10: Cross parachute inflation for wind tunnel experiment. The initial shape is flat cross, the diameter is 1.27 m, the parachute has 20 suspension lines which are 1.27 m each. The simulation starts from a fully folded state and ended when the canopy is opened. The fabric parameters are the same as being used in Fig. 9.

exists an upper bound $|\mu|\le\sqrt{2Mk/m}$, where M is the maximum number of neighbors a spring mass point can have.

The nonlinear spring model is more difficult to analyze. But we found that the force along the direction to the neighbors of a vertex is the same as in the linear model. Numerically, we have showed that indeed, the motion of a spring mass point is oscillatory along the tangential direction of the fabric surface. The motion in the direction normal to the surface is not oscillatory in general. Fourier analysis of the tangential motion on an arbitrary sample point showed that the frequency of the oscillation is bounded by $\sqrt{Mk/m}/2\pi$, where $M\approx 7$.

For the oscillatory motion, the first order Euler forward scheme for the ODE system is increasing in amplitude. Higher order Runge-Kutta scheme is a much efficient way to solve the equations. Our computation showed that the fourth order Runge-Kutta scheme with $\mu_{max}\Delta t \leq 0.1$ gives very stable and accurate solution to the spring mass ODE system.

There are still two open problems. The first is that although the analysis of Model-I through the Gershgorin circle theorem gives an upper bound of the eigen frequency as $\omega_c \leq \sqrt{2Mk/m}$, our numerical tests suggest that this upper bound is not a sharp bound. The minimum bound appears to be $\omega_c \leq \sqrt{Mk/m}$. The second is that we still need the analytical proof of the upper bound for the nonlinear system of Model-II.

Acknowledgments

The authors would like to acknowledge many discussions with Xiangmin Jiao and Keh-Ming Shyue. Joungdong Kim and Xiaolin Li are supported in part by the US Army Research Office under the ARO grant award W911NF0910306. Xiaolin Li would like to thank the Department of Mathematics, National Taiwan University and to acknowledge the generous support from National Science Council of The Republic of China, Grant NSC 101-2811-M-002-006 on his sabbatical visit during which this work is accomplished.

References

[1] R. Fedkiw A. Selle, M. Lentine. A mass spring model for hair simulation. ACM Transactions on Graphics (Proceedings of the ACM SIGGRAPH'08 conference), 27, 2008.

[2] M. Aono, D. Breen, and M. Wozny. Fitting a woven cloth model to a curved surface: mapping algorithms. Computer-Aided Design, 26(4):278–292, April 1994.

[3] M. Aono, P. Denti, D. Breen, and M. Wozny. Fitting a woven cloth model to a curved surface: dart insertion. IEEE Computer Graphics and Applications, 16(5):60–70, September 1996.

[4] U. Ascher and E. Boxerman. On the modified conjugate gradient method in cloth simulation. The Visual Computer, 19(7–8):523–531, December 2003.

[5] D. Baraff and A. Witkin. Large steps in cloth simulation. In Proceedings of ACM SIGGRAPH 98, pages 43–54. ACM Press, 1998.

[6] R. Bargmann. Real time cloth simulation. Master's thesis, Ecoly Polytechnique Fédérale de Lausanne (EPFL) / Eidgenössische Technische Hoschschule Zürich (ETHZ), Lausanne / Zurich, Switzerland, 2003.

[7] H. Bez, A. Bricis, and J. Ascough. A collision detection method with applications in CAD systems for the apparel industry. Computer-Aided Design, 28(1):27–32, 1996.

[8] W. Bo, X. Liu, J. Glimm, and X. Li. Primary breakup of a high speed liquid jet. ASME Journal of Fluids Engineering, submitted, 2010.

[9] D. Breen. A particle-based model for simulating the draping behavior of woven cloth. PhD thesis, Rensselaer Polytechnic Institute, 1993.

[10] D. Breen, D. House, and M. Wozny. Predicting the drape of woven cloth using interacting particles. In Proceedings of ACM SIGGRAPH 94, pages 365–372. ACM Press, 1994.

[11] R. Bridson, S. Marino, and R. Fedkiw. Simulation of clothing with folds and wrinkles. In Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA 2003), pages 28–36. ACM Press, 2003.

[12] Robert Bridson, Ronald Fedkiw, and John Anderson. Robust treatment of collisions, contact and friction for cloth animation. ACM Transactions on Graphics, 21:594–603, 2002.

[13] J. C. Butcher. The numerical analysis of ordinary differential equations : Runge-Kutta and general linear methods. 1987.

[14] M. Carignan, Y. Yang, N. Magnenat-Thalmann, and D. Thalmann. Dressing animated synthetic actors with complex deformable clothes. In Computer Graphics (Proceedings of ACM SIGGRAPH 92), pages 99–104. ACM Press, 1992.

[15] K.-J. Choi and H.-S. Ko. Stable but responsive cloth. ACM Transactions on Graphics, 21:604–611, 2002.

[16] Jian Du, Brian Fix, James Glimm, Xicheng Jia, Xiaolin Li, Yunhua Li, and Lingling Wu. A simple package for front tracking. J. Comput. Phys., 213:613–628, 2006.

[17] B. Eberhardt, M. Meißner, and W. Straßer. Knit fabrics. In D. House and D. Breen, editors, Cloth Modeling and Animation, pages 123–144. A.K. Peters, 2000.

[18] B. Eberhardt, A. Weber, and W. Straßer. A fast, flexible, particle-system model for cloth draping. IEEE Computer Graphics and Applications, 16(5):52–59, September 1996.

[19] E. George, J. Glimm, X. L. Li, Y. H. Li, and X. F. Liu. The influence of scale-breaking phenomena on turbulent mixing rates. Phys. Rev. E, 73:016304, 2006.

[20] A. H. Barr J. C. Platt. Constraints methods for flexible models. In SIGGRAPH '88 Proceedings of the 15th annual conference on Computer graphics and interactive techniques, pages 279–288, 1988.

[21] J.-D. Kim, Y. Li, and X.-L. Li. Simulation of parachute FSI using the front tracking method. Journal of Fluids and Structures, 37:101–119, 2013.

[22] X. Provot. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In Proceedings of Graphics Interface (GI 1995), pages 147–154. Canadian Computer-Human Communications Society, 1995.

[23] R. Samulyak, T. Lu, P. Parks, J. Glimm, and X. Li. Simulation of pellet ablation for tokamak fuelling with itaps front tracking. Journal of Physics: Conf. Series, 125:012081, 2008.

[24] D. Terzopoulos and K. Fleischer. Deformable models. The Visual Computer, 4(6):306–331, December 1988.

[25] D. Terzopoulos and K. Fleischer. Modeling inelastic deformation: viscoelasticity, plasticity, fracture. In Computer Graphics (Proceedings of ACM SIGGRAPH 88), pages 269–278. ACM Press, July 1988.

[26] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer. Elastically deformable models. In Computer Graphics (Proceedings of ACM SIGGRAPH 87), pages 205–214. ACM Press, July 1987.

[27] D. Terzopoulos and K. Waters. Analysis and synthesis of facial image sequences using physical and anatomical models. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 15:569–579, 1993.

[28] P. Volino, M. Courchesne, and N. Magnenat-Thalmann. Versatile and efficient techniques for simulating cloth and other deformable objects. In Proceedings of ACM SIGGRAPH 95, pages 137–144. ACM Press, 1995.