Stony Brook University

Modeling of Parachute Dynamics with Front Tracking Method

A Dissertation Presented

by

Joung-Dong Kim

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Applied Mathematics and Statistics

Stony Brook University

December 2012

Stony Brook University

The Graduate School

Joung-Dong Kim

We, the dissertation committee for the above candidate for the
Doctor of Philosophy degree, hereby recommend
acceptance of this dissertation.

Xiaolin Li - Dissertation Advisor
Professor, Department of Applied Mathematics and Statistics

James Glimm - Chairperson of Defense
Professor, Department of Applied Mathematics and Statistics

Xiangmin Jiao - Member
Associate Professor, Department of Applied Mathematics and Statistics

Foluso Ladeinde - Outside Member
Associate Professor, Department of Mechanical Engineering

This dissertation is accepted by the Graduate School.

Charles Taber
Interim Dean of the Graduate School

Abstract of the Dissertation

Modeling of Parachute Dynamics with Front Tracking Method

by

Joung-Dong Kim

Doctor of Philosophy

in

Applied Mathematics and Statistics

Stony Brook University

2012

We use the front tracking method on a spring system to model the dynamic evolution of parachute canopies. The canopy surface of a parachute is represented by a triangulated surface mesh with preset equilibrium length on each side of the simplices. The stretching and wrinkling of the canopy and its supporting string chords (risers) are modeled by the spring system. The spring constants of the canopy and the risers are chosen based on the analysis of Young's modulus for fabric surface and string chord. 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 three spring models and we proved in one case that all eigenvalues are imaginary and there exists an upper bound for eigen-frequency. 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. And also high order method helps to control amplification of system. Damping is added to dissipate the excessive spring internal energy. The current model does not have radial reinforcement cables and has not taken into account the canopy porosity. This mechanical structure is coupled with the incompressible Navier-Stokes solver through the "Impulse method" which computes the velocity of the point mass by superposition of momentum. We analyzed the numerical stability of the spring system and used this computational module to simulate the flow pattern around a static parachute canopy and the dynamic evolution during the parachute inflation process. The numerical solutions have been compared with the available experimental data and there are good agreements in the terminal descent velocity and breathing frequency of the parachute.

Key Words: front tracking, spring model, eigen frequency, parachute inflation

To my father,
Gil-Yeon Kim

Table of Contents

List of Figures

List of Tables

Acknowledgements

Even though only my name appears on the cover of this dissertation, a great number of people have contributed to its production. I owe my gratitude to all those people who have made this dissertation possible and because of these people, this great experience has been one that I will cherish forever.

I would like to express my sincere gratitude to my advisor Professor Xiaolin Li for his continuous support of my Ph.D. study. It has been honor and privilege to have worked under the direction of Professor Xiaolin Li. Through his excellent guidance, care, and patience, I have grown as a student and became an applied mathematician who has knowledge in mathematics, physics, and computer science. He has helped me to have much better understanding in everything. He gave me the courage to develop and express myself professionally. That made me grow as a student; accomplishing a lot during my graduate study years. He is a lifetime role model for me.

I am deeply indebted to the support of Professor James Glimm. I have learned lots of important scientific and mathematical knowledge, and his insightful comments and constructive criticisms at different stages of my research were thought provoking and they helped me focus my ideas. And thanks to Professors Xiangmin Jiao, Roman Samulyak who provided encouragement and

valuable technical knowledge.

I would also like to thank Professor Foluso Ladeinde for being on my dissertation committee.

I would like to thank all my friends during my years of study as a graduate student at Stony Brook for their friendship and encouragement. I would like to mention Dr. HyunKyung Lim, Dr. Tulin Kaman, Dr. Shuqiang Wang, Ryan Kaufman, Yan Li, Yijing Hu, Qiangqiang Shi, and Saurabh Joglekar. They have shared with me many interesting and inspiring ideas. And also I am happy to mention Hyuck Hur who is my good friend and tennis partner in Stony Brook.

I would like to acknowledge the support of Department of Applied Mathematics and Statistics and its staff. Special thanks to Janice Hackney for her kindness, friendship and support, together with the other staff.

Last but not the least; I would like to thank my family. I wanted to make my father's dream come true through my life. I hope my Ph.D. degree will be a great gift for my father with my upmost respect. I can't express my gratitude for my mother in words, whose unconditional love has been my greatest strength. Without her love and wisdom, I wouldn't have been able to accomplish all that I have so far. As for my lovely wife Grace Lee, who has been there cheering me on and has stood by me through good times and bad, I give a special thanks from the bottom of my heart. You are my inspiration, love and life. And my little daughter, Allison Yeoleum Kim gives me numerous, unexpected and awesome ideas for my life. When she was born, she had changed my life fantastically. I could not have accomplished this without the

support of my parents and family. My dissertation is dedicated to my parents, wife and daughter.

Above all, I thank God for his grace, wisdom, favour, faithfulness, and protection.

Chapter 1

Introduction

1.1 Overview and Motivation

Simulation of parachute inflation *via* computational method has attracted attention of scientists at Laboratories of Department of Defense and academic alike. Some of the most successful studies include those by Stein, Benney, *et al.*[56, 55, 52, 57, 53, 54] using the finite element method for the fluid and structure dynamics. The study of Tezduyar *et al.*[65, 64, 60, 59, 66] focused on the fluid structure interaction of the parachute system. By applying the Deforming-Spatial-Domain/Stabilized Space-Time (DSD/SST) method, they have successfully addressed the computational challenges in handling the geometric complexities and the contact between parachutes in a cluster. Kim and Peskin *et al.*used the immersed boundary method to study the semi-opened parachute in both two and three dimensions [44, 45], their simulations are for small Reynold number (about 300) and applied payload with several grams. Yu and Min [71] studied the transient aerodynamic characteristics of the parachute opening process. Karagiozis used the large-eddy simulation to

study parachute in Mach 2 supersonic flow [41]. Purvis [48, 49] used springs to represent the structures of forebody, suspension lines, canopy, etc. In these papers, the author used cylindrical coordinate with the center line as the axis. In a paper by Strickland *et al.*[58], the authors developed an algorithm called PURL to couple the structure dynamics (PRESTO) and fluid mechanics (CURL), in which mass is added to each structure node based on the diagonally added mass matrix and a pseudo is computed from the fluid code which is the sum of the actual pressure and the pressure associated with the diagonally added mass. Tutt and Taylor [68, 67] simulated the parachute through the LS-DYNA code. They used an Eulerian-Lagrangian penalty coupling algorithm and multi-material ALE capabilities with LS-DYNA to replicate the inflation of small round canopies in a water tunnel.

Fabric material belongs to flexible objects and is more difficult to model than rigid objects. 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.

Many authors have contributed to the modeling of cloth and fabric sur-

face. Terzopoulos and Fleischer [63, 61, 62] 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.*[11, 12], presented a particle-based model capable of being tuned to reproduce the static draping behavior of specific kinds of woven cloth. Eberhardt *et al.*[26, 25] 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 [17] also used particle spring model, our model is base on their work. On the physics based modeling, Platt and Barr[36] 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.*[16] discussed the use of physics-based models for animating clothes on synthetic actors in motion. Provot [47] described a physics-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.*[70] presented an efficient set of techniques that simulates any kind of deformable surface in various mechanical situations. Hsiao and Chen [35] used the spring model to draw the cloth pattern. They found that cloth shows different appearance with different values of the spring constant. Aileni *et al.*[1] applied the mass-spring

model for a three dimensional simulation of apparel products using virtual mannequins. Their model supports different types of human bodies created in the virtual environment. Ji [37] used the mass-spring model to describe the dynamic draping behavior of the selected five types of fabric materials including woven and knitted fabrics. In his paper, the material properties are measured through the Kawabata Evaluation System (KES) [42].

1.2 Front Tracking Methods

Front Tracking is a numerical algorithm which assigns special degrees of freedom to a surface, moving dynamically through a background grid. This method, when coupled with underlying PDE solvers, can provide high resolution to the geometry and physical variables across the interface. Front Tracking is a Lagrangian interface method; it can deliver solutions superior to Eulerian methods, including the level set and the volume of fluid methods [21].

1.3 Fabric Dynamics

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.

1.4 Parachute Dynamics

We report a computational study of parachute inflation through application of the front tracking method, a method which has been successfully applied to many other problems in physics and engineering [24, 33, 46, 51]. The front tracking method has been coded in a software library named *Fron Tier*. We modify the data structure to allow the application of the *FronTier* library to the dynamic motion of fabric surface driven by the gravitational force of payload and the fluid pressure. We discretize the fabric surface in a homogeneously triangulated surface mesh. One of the important properties of a fabric surface is that such a surface has finite stretchability. This property is mathematically realized by modeling the surface as a spring system using the vertices as mass points and the triangle sides as springs. A finite friction force is added to dissipate the spring internal energy and to prevent over-excitation of the system. The spring system is coupled with a finite difference solver for the incompressible Navier-Stokes equation. Our simulations extend to the field range of parachute dimensions and are compared with the realistic payload of certain types of parachutes.

1.5 Dissertation Organization

In Chapter 2, we will introduce the mathematical models and the numerical method in detail and discuss some of the major modifications we have made to both the *FronTier* functionalities and its coupling with the incompressible fluid solver. Chapter 3 presents the motion of spring system, especially eigenfrequency will be discussed. Chapter 4 discusses the numerical accuracy and stability in the computation of the spring system. In Chapter 5, we present several benchmark test cases on the modeling of fabric surface through the front tracking method and in Sec. 5.6, we report the simulation on dynamic motion of three types of parachutes: the T-10 personnel parachute, the G-11 cargo parachute, and the cross parachute. The mathematical model we used in this paper has certain simplifications, to make more realistic simulation of the parachute system, some additional physical and numerical components must be added. In Chapter 6 the numerical solutions have been compared the available experimental data for validation. Chapter 7 will discuss some of the on-going work and amendment to our current computational method for the parachute system.

Chapter 2

Mathematical Model and Numerical Method

2.1 Navier-Stokes equations

Since the focus of this study is on incompressible flow, the fluid equations we use to model the air flow as a continuum medium are the incompressible Navier-Stokes equations, which in vector form can be written as

$$\rho\frac{D\mathbf{u}}{Dt} + \nabla p = \mu\nabla^2\mathbf{u} + \mathbf{g}, \tag{2.1}$$

where $\frac{D}{Dt} = \dfrac{\partial}{\partial t} + \mathbf{u}\cdot\nabla$ is the total derivative of the fluid. The incompressibility is described by the divergence-free condition:

$$\nabla\cdot\mathbf{u} = 0. \tag{2.2}$$

For the parachute system, this equation is solved through the projection method [15] with special treatment at the canopy surface.

2.2 Projection method

Projection method is an effective means of numerically solving time-dependent incompressible fluid flow problems. The advantage of the projection method is that the computation of velocity and the pressure fields are decoupled. Some approximation to the momentum Eq. (2.1) is used to determine the velocity u or a provisional velocity, and then an elliptic equation is solved that enforces the divergence constraint Eq. (2.2) and determines the pressure. In some variations, the viscous term in Eq. (2.1) is advanced in a separate step from the advective terms. The original projection method is that the velocity field is forced to satisfy a discrete divergence constraint at the end of each time step. Projection methods which enforce a discrete divergence constraint, or exact projection methods, have often been replaced with approximate projection methods. Approximate projection methods are used because of observed weak instabilities in exact methods and the desire to use more complicated or adaptive finite difference meshes on which exact projections are difficult or mathematically impossible to implement. Additionally, as with all fractional step methods, a crucial issue is how boundary conditions are determined for some or all of the intermediate variables.

Projection method pioneered by Chorin [18, 19] for numerically integrating Eqs. (2.1) and (2.2) is based on the observation that the left-hand side of equation Eq. (2.1) is a Hodge decomposition. Hence an equivalent projection formulation is given by

$$u_t = P[-(u \cdot \nabla)u + \nu \nabla^2 u], \tag{2.3}$$

where P is the operator which projects a vector field onto the space of divergence-free vector fields with appropriate boundary conditions. In the 1980s, several papers appeared in which second-order accurate versions of the projection method were proposed. Those of Goda [34], Bell *et al.*[8], Kim and Moin [43], and Van Kan [69] are motivated by the second-order, time-discrete semi-implicit forms of Eqs. (2.1) and (2.2),

$$\frac{u^{n+1} - u^n}{\Delta t} + \nabla p^{n+1/2} = -[(u \cdot \nabla)u]^{n+1/2} + \frac{\nu}{2}\nabla^2(u^{n+1} + u^n) \tag{2.4}$$

$$\nabla \cdot u^{n+1} = 0, \tag{2.5}$$

where $[(u \cdot \nabla)u]^{n+1/2}$ represents a second-order approximation to the convective derivative term at time level $t^{n+1/2}$ which is usually computed explicitly. Spatially discretized versions of the coupled Eqs. (2.4) and (2.5) are cumbersome to solve directly. Therefore, a fractional step procedure can be used to approximate the solution of the coupled system by first solving an analog to Eq. (2.4) for an intermediate quantity u^*, and then projecting this quantity onto the space of divergence-free fields to yield u^{n+1}. In general this procedure is given by

Step 1: Solve for the intermediate field u^*

$$\frac{u^* - u^n}{\Delta t} + \nabla q = -[(u \cdot \nabla)u]^{n+1/2} + \frac{\nu}{2}\nabla^2(u^* + u^n), \tag{2.6}$$

$$B(u^*) = 0, \tag{2.7}$$

where q represents an approximation to $p^{n+1/2}$ and $B(u^*)$ a boundary

condition for u^* which must be specified as part of the method.

Step 2: Perform the projection

$$u^* = u^{n+1} + \Delta t \nabla \phi^{n+1} \tag{2.8}$$

$$\nabla \cdot u^{n+1} = 0, \tag{2.9}$$

using boundary conditions consistent with $B(u^*) = 0$ and $u^{n+1}|_{\partial\Omega} = u_b^{n+1}$.

Step 3: Update the pressure

$$p^{n+1/2} = q + L(\phi^{n+1}), \tag{2.10}$$

where the function L represents the dependence of $p^{n+1/2}$ on ϕ^{n+1}. Once the time step is completed, the predicted velocity u^* is discarded, not to be used again at that or later time steps. There are three choices that need to be made in the design of such a method. They are the pressure approximation q, the boundary condition $B(u^*)$, and the function $L(\phi^{n+1})$ in the pressure-update equation. An important issue is that the boundary condition for u^* must be consistent with Eq. (2.8) although at the time the boundary conditions are applied the function ϕ^{n+1} is not yet known and hence must be approximated.

In the first step of the method, if q is a good approximation to $p^{n+1/2}$, the field u^* may not differ significantly from the fluid velocity and thus a reasonable choice for the boundary condition $B(u^*) = 0$ may be $(u^* - u_b)|_{\partial\Omega} = 0$. On the other hand, one may not be interested in computing

the pressure at every time step and would like to choose $q = 0$ and obviate the third step in the method. In this case u^* may differ significantly from the fluid velocity, requiring the boundary condition $B(u^*)$ to include a nontrivial approximation of $\nabla\phi^{n+1}$ in Eq. (2.8). Regarding the third step of the method, substituting Eq. (2.8) into Eq. (2.6), eliminating u^*, and comparing with Eq. (2.4) yield a formula for the pressure-update

$$p^{n+1/2} = q + \phi^{n+1} - \frac{\nu\Delta t}{2}\nabla^2\phi^{n+1}. \tag{2.11}$$

The last term of this equation plays an important role in computing the correct pressure gradient and allows the pressure to retain second-order accuracy up to the boundary. Without this term, the pressure gradient may have zeroth-order accuracy at the boundary even if the pressure itself is high-order accurate.

2.3 Structure Component

The non-fluid material in the parachute simulation is called the structure component whose motion is governed by the Newton's second law subject to certain internal constraints. Both the canopy surface and the string chords (or the risers) which connect the canopy and the payload are flexible structures and they too, are continuum systems. In many literatures, the motion of the structure is described by the quasi-ordinary equation

$$\rho_i \frac{d^2\mathbf{x_i}}{dt^2} = \mathbf{f_i} - \nu\frac{d\mathbf{x_i}}{dt} + \nabla\cdot\sigma_\mathbf{i}, \tag{2.12}$$

where at position $\mathbf{x_i}$, ρ_i is the density, $\mathbf{f_i}$ is the external force density (for example, due to gravity and fluid pressure), $\sigma_\mathbf{i}$ is the stress tensor, and ν is the damping coefficient. In general, all derivatives in Eq. (2.12) should be considered as partial derivatives. However, very few have attempted to solve Eq. (2.12) exactly. Like the fluid, discretization of Eq. (2.12) is also needed. In the model we use for this paper, we seek to approximate Eq. (2.12) through physical intuition, that is, we approximate each discretized element as a mass point while the stress tensor $\sigma_\mathbf{i}$ is approximated by a set of springs connecting to the neighboring points. The spring system has only the restoring force against stretching and compression, therefore it may not exactly describe the structure system, especially when the structure's stress tensor may include restoring force against bending and twisting. Since the structure involved in the parachute study contains only fabric surface and string chord, we believe such approximation is good enough and can capture the most important properties of the structure dynamics in the parachute system. The details of such system will be described in the following section. To correctly model the parachute system, an accurate coupling between the Navier-Stokes equation and the structure dynamics must be carefully considered near the canopy surface. The method we designed for the simulations in this paper uses the superposition of impulse on every mass point. Each mass point in the spring system acts as an elastic boundary point and exerts an impulse to the fluid in its normal direction. Our algorithm ensures that the action and reaction between the spring mass point and the fluid solver are equal in magnitude and opposite in directions, a requirement of Newton's third law. The numerical

detail of the algorithm is described in Sec. 2.9.

2.4 Numerical Methods

The computational procedures for the parachute system are demonstrated by the flow-chart in Fig. 2.1. The numerical method we have used in this paper for the simulation of parachute system contains several components. The data structures and many functionalities are based on the *FronTier* library developed for the front tracking method. The parachute canopy is modeled by the spring system on a homogeneously triangulated mesh while the string chords connecting the payload. A finite damping coefficient is added to both the canopy and the riser spring chains in order to dissipate the kinetic energy of oscillatory motion. The fluid flow is computed by solving the incompressible Navier-Stokes equation using finite difference method with high order coupling at the boundary and interior interface. To simplify the interaction between the canopy and the fluid and to avoid harmful damping to 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 force each point mass receives from its neighboring points under the spring model. Most of the components in the numerical model are formally in at least the second order, but due to its geometrical complexity, the overall system cannot achieve global second order due to splitting computation over each system. The resulting equation 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 sections, 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 non-linear ODE system, many researchers have tried to use implicit method. For example, Baraff *et al.*[6] described a cloth simulation system that can stably take large time steps. It introduces an implicit integration method and applies on a triangular 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 leads 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.

2.5 Representation of Fabric Structure

Using a spring-mass system to model a fabric surface has been explored by computer scientists and applied mathematicians. This spring system provides

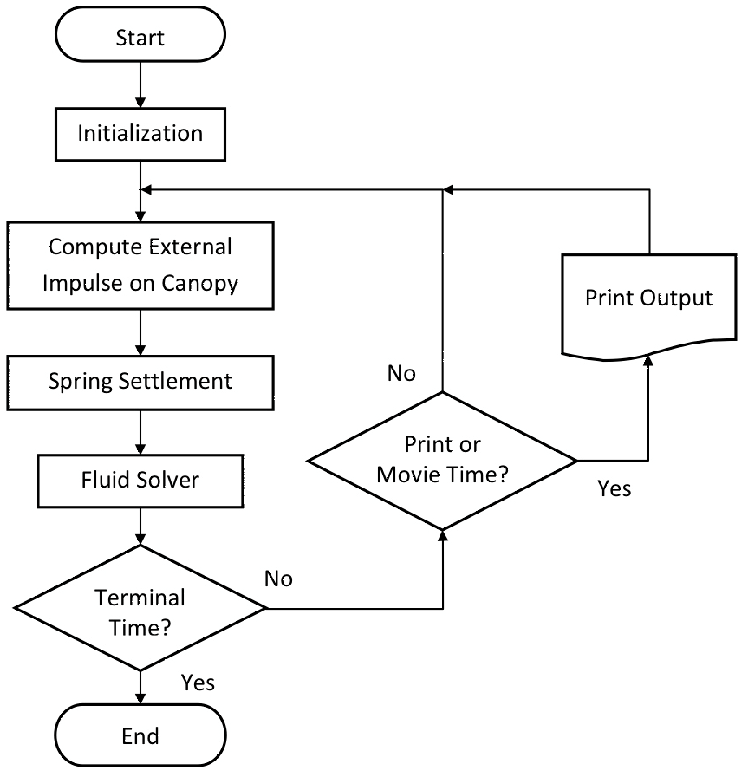

Figure 2.1: The flow chart of the computation for the parachute system using *FronTier* as the base library. The spring system is a new module built on the *FronTier* data structure and functions. The system is coupled with a high order incompressible solver for the Navier-Stokes equation.

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 by Choi and Ko [17] 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. The front tracking library contains data structure and

functionalities to optimize and resolve the hyper-surface mesh with topological consistency. This method has been used for the simulations of fluid interface instabilities [24, 23, 29, 28], diesel jet droplet formation [10], and plasma pellet injection process [50, 51]. In these problems, the hyper-surface is used to model the interior discontinuities of materials and such manifold surface may undergo complicated changes in geometry and topology. The modeling of fabric surface is simpler in topological handling due to the fact that a fabric surface cannot bifurcate. 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 revisions 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. 2.2 illustrates the spring model on the triangulated mesh using the front tracking data structure.

The second difference between the material interface and the fabric sur-

face 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.

2.6 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

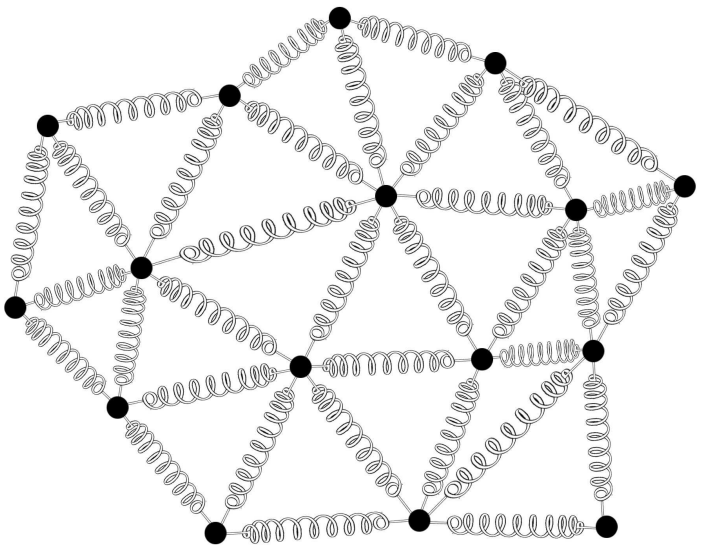

Figure 2.2: Fabric surface is modeled by a homogeneously triangulated mesh using the *FronTier* data structure. The edges of each triangle have preset equilibrium length. The dynamic motion such as stretching and wrinkling of the fabric surface is modeled by solving the system of ordinary equations for the spring system.

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

$$V_{ij} = \frac{k}{2}\left|(\mathbf{x}_i - \mathbf{x}_j) - (\mathbf{x}_{i0} - \mathbf{x}_{j0})\right|^2, \tag{2.13}$$

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. (2.13) 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 chapter, 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

$$V_{ij} = \frac{k}{2}(|\mathbf{x}_i - \mathbf{x}_j| - l_{ij}^0)^2, \tag{2.14}$$

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,

$$L = T - V = \sum_{i=1}^{N} \frac{1}{2} m |\dot{\mathbf{x}}_i|^2 - \frac{1}{4} \sum_{i=1}^{N} \sum_{j=1}^{N} k \left| (\mathbf{x}_i - \mathbf{x}_j) - (\mathbf{x}_{i0} - \mathbf{x}_{j0}) \right|^2 \eta_{ij} \quad (2.15)$$

for Model-I and

$$L = T - V = \sum_{i=1}^{N} \frac{1}{2} m |\dot{\mathbf{x}}_i|^2 - \frac{1}{4} \sum_{i=1}^{N} \sum_{j=1}^{N} k (|\mathbf{x}_i - \mathbf{x}_j| - l_{ij}^0)^2 \eta_{ij} \quad (2.16)$$

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.

Applying the 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 $\mathbf{x}_i$ where $q = (\mathbf{x}, t)$, we have

$$m \frac{d\dot{\mathbf{x}}_i}{dt} = m \frac{d^2 \mathbf{x}_i}{dt^2} = - \sum_{j=1}^{N} \eta_{ij} k \left((\mathbf{x}_i - \mathbf{x}_j) - (\mathbf{x}_{i0} - \mathbf{x}_{j0}) \right), \quad (2.17)$$

for Model-I and

$$m \frac{d\dot{\mathbf{x}}_i}{dt} = m \frac{d^2 \mathbf{x}_i}{dt^2} = - \sum_{j=1}^{N} \eta_{ij} k \left(\mathbf{x}_i - \mathbf{x}_j - l_{ij}^0 \mathbf{e}_{ij} \right), \quad (2.18)$$

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 $\mathbf{x}_j$.

2.7 The Forces in the Two Models

To see the difference between the two spring models, we rewrite the right hand side of Eq. (2.17) as follows

$$\begin{aligned} m\frac{d^2\mathbf{x}_i}{dt^2} &= -\sum_{j=1}^{N}\eta_{ij}k\left(\mathbf{x}_i-\mathbf{x}_j-l_{ij}^0\mathbf{e}_{ij}+l_{ij}^0\mathbf{e}_{ij}-\mathbf{x}_{i0}+\mathbf{x}_{j0}\right) \\ &= -\sum_{j=1}^{N}\eta_{ij}k\left((|\mathbf{x}_i-\mathbf{x}_j|-l_{ij}^0)\mathbf{e}_{ij}+l_{ij}^0(\mathbf{e}_{ij}-\mathbf{e}_{ij}^0)\right). \end{aligned} \tag{2.19}$$

We may use $\mathbf{f}_i^s$ and $\mathbf{f}_i^r$ to represent the two forces in the right hand side of Eq. (2.19), 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:

$$m\frac{d^2\mathbf{x}_i}{dt^2} = \mathbf{f}_i^r, \tag{2.20}$$

and since Model-II is

$$m\frac{d^2\mathbf{x}_i}{dt^2} = \mathbf{f}_i^s, \tag{2.21}$$

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

$$m\frac{d^2\mathbf{x}_i}{dt^2} = \mathbf{f}_i^s + \mathbf{f}_i^r, \tag{2.22}$$

Eq. (2.20) refers to bending only, Eq. (2.21) refers to stretching only (fabric), while Eq. (2.22) 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 chapter, 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.

2.8 Fluid Solver

The calculation of advection of the velocity field is based on the fifth order finite difference WENO schemes by Jiang and Shu[38] with a general framework for the design of smoothness indicators and nonlinear weights. A key component of the WENO scheme is the linear combination or reconstruction of lower order fluxes to obtain a higher order approximation. The WENO schemes use the idea of adaptive stencils to automatically achieve high order accuracy and non-oscillatory property near discontinuities. In the system case, WENO scheme is based on local characteristic decomposition and flux split-

ting to avoid spurious oscillation. The time discretization of WENO schemes is implemented by a class of high order TVD Runge-Kutta methods. Assuming $\mathcal{L}(\mathbf{u})$ is a discretization of the spatial advection operator, the third-order TVD Runge-Kutta is

$$\begin{aligned}
\mathbf{u}^{(1)} &= \mathbf{u}^n + \Delta t \cdot \mathcal{L}(\mathbf{u}^n) \\
\mathbf{u}^{(2)} &= \frac{3}{4}\mathbf{u}^n + \frac{1}{4}\mathbf{u}^{(1)} + \frac{1}{4}\Delta t \cdot \mathcal{L}(\mathbf{u}^{(1)}) \\
\mathbf{u}^{n+1} &= \frac{1}{3}\mathbf{u}^n + \frac{2}{3}\mathbf{u}^{(2)} + \frac{2}{3}\Delta t \cdot \mathcal{L}(\mathbf{u}^{(2)}).
\end{aligned}$$

The advection equation is a scalar equation but has both linear and nonlinear flux functions (flux of Burgers equation). Using the flux version of the WENO scheme, this can be computed robustly. This is followed by a second order Crank-Nicolson solver for the diffusion (viscous) term. The fluid pressure is a derived variable from the elimination of velocity divergence at each time step. Only pressure gradient or pressure difference (across canopy) is used to calculate the force which the fluid interacts with the structure (canopy).

2.9 The Fluid-Canopy Coupling

For incompressible Navier-Stokes equation, it is the boundary condition that controls the dynamics of the solution. For the parachute problem, the boundary condition consists of two parts, the external boundary and the two interior sides of the canopy surface. In our computation, we have three different types of external boundary conditions, the preset Dirichlet boundary, the flow-through boundary, and the periodic boundary. The periodic bound-

ary does not need special treatment and always follows the order of the numerical scheme. The Dirichlet boundary for the hyperbolic and parabolic (advection-diffusion) part of the numerical scheme is also relatively easy to specify. Normally the preset Dirichlet boundary is on the upwind side while the flow-through boundary is on the downwind side. The only approximation we made is to assume the downwind side of flow-through boundary is a constant extrapolation. This is needed for the parabolic equation. To compute the projection, we have used Neumann boundary for the preset Dirichlet boundary and the constant pressure (or ϕ) for the flow-through boundary. Since only the pressure gradient is physically significant, without losing generality, we set the ϕ at the flow-through boundary to zero. The interaction between fluid and the canopy is the most crucial part of the algorithm for the parachute simulation. We noted that the immersed boundary method by Kim and Peskin [44, 45] can only carry payload up to a few grams. This is not in the realistic range of parachute payload. The T-10 personnel parachute and the G-11 cargo parachute both carry payload ranging from hundreds to thousands of kilograms. Aside from the fact that Kim and Peskin's simulations are in fluid with small Reynold number, we also think that a more accurate modeling between the air flow and the canopy surface should be considered.

The system described by Eq. (2.17) 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

$$\mathbf{I}_i = \mathbf{I}_i^e + \mathbf{I}_i^s. \tag{2.23}$$

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

Physically, the canopy experiences three forces, the gravitational force due to the weight of the fabric, the lift force due to the pressure difference between the two sides of the canopy, and the internal force, which in our model is the spring restoring force and the friction force (to prevent the spring system becoming over-excited). The gravitational force of the payload is propagated through the spring system from the string chord to the boundary of the canopy, and then spread to each mass point of the canopy through the elastic sides of simplices. Although the interaction between the fluid and canopy has the participation of both the external and the internal forces, for each mass point in the spring system, we can still divide the impulse into three components,

the gravitational impulse, the fluid impulse due the pressure difference between the two sides of the canopy, and the internal impulse due to its neighboring mass points in the spring system, that is

$$\mathbf{I}_i^c = \mathbf{I}_{gi}^c + \mathbf{I}_{pi}^c + \mathbf{I}_{si}^c \tag{2.24}$$

Our current model has not considered the fluid interaction with the mass point of the string chord and the payload. Therefore for these mass points, the impulse is

$$\mathbf{I}_i^s = \mathbf{I}_{gi}^s + \mathbf{I}_{si}^s. \tag{2.25}$$

In our method, the external impulse (due to gravity and pressure) is time integrated for each mass point, that is

$$\mathbf{I}_{gi} = \int_0^t m\mathbf{g}dt \tag{2.26}$$

for both canopy and string chord mass points and

$$\mathbf{I}_{pi} = \int_0^t \sigma(p^- - p^+)\mathbf{n}dt \tag{2.27}$$

for canopy points only, where p^- and p^+ are the pressure on lower and upper sides of the canopy, σ is the mass density of canopy per unit area, and $\mathbf{n}$ is the unit normal vector pointing from lower to upper side of the canopy. We would like to emphasize that the current calculation of fluid impulse has considered only the pressure in the normal direction, we have followed Kalro and Tezduyar [40] for using this simplification. A more accurate fluid interaction should

include the velocity shear near the canopy surface and the stress to the surface. We will put this as future improvement in the new papers.

The internal impulse for both canopy and string chord mass points are solved by the damping spring system. The impulse due to payload force is propagated through the string chords to the edge points of the canopy surface.

The interaction between the canopy and fluid is through the normal velocity component of each mass point on the canopy. 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 canopy 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 $\mathbf{v}_i^s = \mathbf{I}_i^s/m$.

We have studied two ways to implement the reacting impulse which the canopy exerts on the fluid (or vice versa). The first method is through the

immersed boundary method which treats the reaction of the canopy as the normal force approximated by the smoothed delta function. The second method is to use the canopy as an internal moving boundary with a normal velocity (after spring settlement) of the canopy as the boundary value.

In the first method, we compute the increment of the impulse at each point on the canopy. We have followed Peskin's delta function method, that is, let

$$\mathbf{f}(\mathbf{x}, t) = \int \mathbf{F}(s, t)\delta(\mathbf{x} - \mathbf{X}(s, t))ds. \tag{2.28}$$

The difference between our method and Kim and Peskin's method lies in the calculation of $\mathbf{F}$. Instead of computing the tension through the derivative with respect to the arc length, we use the impulse of the mass point as a result of the superposition of three forces from the spring system, that is

$$\mathbf{F}(\mathbf{x}_i, i) = d(\mathbf{I}_g + \mathbf{I}_p + \mathbf{I}_s)/dt \tag{2.29}$$

Eq. (2.29) is more physically realistic, especially because $\mathbf{I}_s$ is solved from the spring equations. In the canopy spring system, we have observed that the tension is high at the top of the canopy where the curvature is almost zero.

The numerical implementation of Eq. (2.29) is straight forward after we have obtained the solution from the ODEs of the spring system. The surface force is the product of normal component of the acceleration and mass density at the canopy surface

$$\mathbf{F}(s, t) = \rho_c(\mathbf{a} \cdot \mathbf{n})\mathbf{n}, \tag{2.30}$$

where $\mathbf{n}$ is the unit normal vector on the canopy surface, and ρ_c is the mass density of canopy per unit area. In the second method we treat the canopy as a moving boundary rather than an immersed interface. Instead of using the normal component of the acceleration as a singular force, we solve a Dirichlet boundary problem on each side of the canopy, we use the normal velocity computed from the ODEs of the spring system. Since the acceleration is the derivative of velocity in a time step, the two methods are physically consistent but with different truncation errors.

2.10 Collision Handling

Collision handling is a challenging work for many researchers in fabric modeling. Recently many authors have presented their algorithms and techniques on this tedious but important work. Bez *et al.*[9] introduced the topological mapping approach to collision detection for the drape simulation work. Bridson *et al.*[14, 13] presented an algorithm to efficiently and robustly process collisions, contact and friction in cloth simulation. Bargmann [7] 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 [22]. 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 $O(nN)$, where n is the number of triangles hashed by one grid block and N is the total number of triangles.

2.11 Local Index Coating Algorithm

Front tracking method has been mostly used for the study of fluid interface problems such as the Rayleigh-Taylor instability [32, 30], Richtmyer-Meshkov instability [27], and the jet problem [33, 31]. In these problems, the fluid interface is topologically a manifold, that is, the two sides of each surface are the boundaries of separate subdomains. Many front tracking functions are based on this assumption. However, in the parachute system, the canopy surface has an open boundary. In general, a space point with a finite distance away from the canopy surface cannot distinguish to which side of the canopy it belongs. But we may still assign the side to which a point belongs if the point is sufficiently close to the canopy surface.

The local side information of a space point close to canopy surface plays an important role in the calculation of pressure difference, and thus the drag force of the air to the canopy. The pressure in one side at the canopy surface is not continuous and therefore should be interpolated and computed using the value from its own side. This is realized by "painting" the grid cells using the so-called locally mesh coating algorithm.

Assuming the domain in which the canopy surface is immersed is indexed by l, the local index coating algorithm follows three steps:

(1). For any grid point P with a distance $d \leq h$ away from the surface, where

h is the grid spacing, find the nearest point on the interface (*FronTier* is well equipped with these geometry functions) P_s. Using the sign of the scalar product $\overline{PP_s} \cdot \mathbf{n_s}$, we can determine the side of the point. Reassign domain index to $l-1$ if the point P is on the negative side, otherwise reassign the index to $l+1$.

(2). Reassign any grid point adjacent to a point indexed $l-1$ to the same; reassign any grid point adjacent to a point indexed $l+1$ to the same. Repeat this assignment for three sweeps. The resulting landscape of the grid indices will look like Fig. 2.3.

(3). The interpolation of side-sensitive variables will use grid points of the same locally coated index.

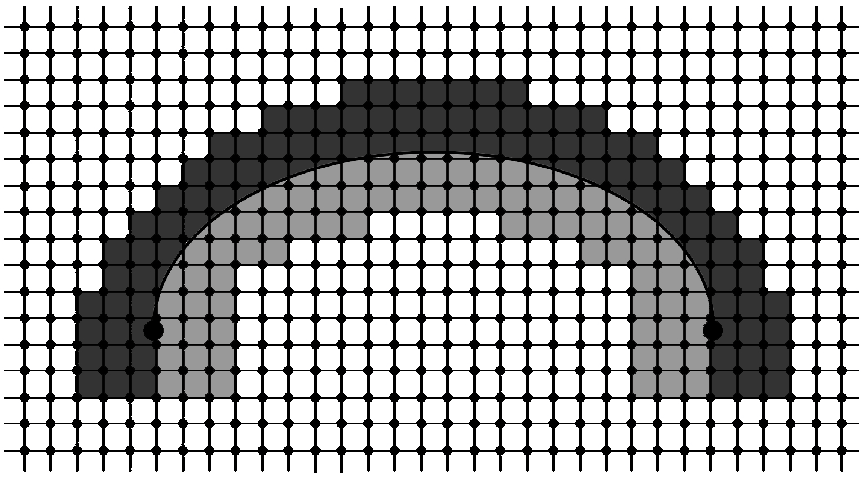

Figure 2.3: The parachute canopy is an open surface and cannot separate the space into subdomains. But we can still coat different indices for mesh cells close to the surface using the local geometrical information. The light and dark shaded polygons represent the sets of mesh cells on the positive and negative sides of the canopy respectively. An interpolation is carried out on vertices of the same color.

Chapter 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

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

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

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

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

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

and

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

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 $\det(\mathbf{A} - \lambda\mathbf{I}_{6N}) = 0$.

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

Or equivalently

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

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 $\mathbf{A}$, it corresponds to the eigenvalue μ of $\mathbf{B}$ by $\lambda = \sqrt{-\mu}$. Now what we want to prove is all the eigenvalues of $\mathbf{B}$ are 0 or positive real numbers. Since $\mathbf{B}$ is symmetric, all the eigenvalues of $\mathbf{B}$ are real numbers. If a real number $\mu < 0$ is an eigenvalue of $\mathbf{B}$, 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 $\mathbf{B}$ are real numbers and they are 0 or positive. Since $\lambda = \sqrt{-\mu}$, we come to the conclusion that all the eigenvalues of $\mathbf{A}$ 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

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

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. (2.18) as follows

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

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. (2.18) as

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

where

$$a_{ij} = \eta_{ij} k \left(1 - \frac{l_{ij}}{r_{ij}}\right), \quad r_{ij} = \left|\mathbf{x}_i - \mathbf{x}_j\right|.$$

We linearize Eq. (3.9) and obtain

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

Using

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

and

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

we obtain

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

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

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

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

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

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

Chapter 4

Numerical Stability of Spring System

The spring system is an energy conservative system if we set the damping coefficient equal to zero. In one dimension or for Model-I (Eq. (2.17)), 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

$$\frac{d\mathbf{w}}{dt} = \Lambda\mathbf{w} \tag{4.1}$$

Therefore to consider the stability of the system Eq. (4.1), we should consider the scalar equation

$$\frac{dy}{dt} = \lambda y. \tag{4.2}$$

The Euler method for Eq. (4.2)

$$y^{n+1} = y^n(1 + \lambda\Delta t) \tag{4.3}$$

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

$$\frac{dy}{dt} = i\omega y, \tag{4.4}$$

with $\omega = |\lambda| = \sqrt{\mu}$. If we use the Euler forward method for Eq. (4.4), that is

$$y^{n+1} = y^n(1 + i\omega\Delta t), \tag{4.5}$$

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

$$\begin{aligned} y^* &= y^n(1 + i\omega\Delta t) \\ y^{n+1} &= y^n + \tfrac{1}{2}i\omega\Delta t(y^n + y^*) = \left(1 - \tfrac{1}{2}\omega^2\Delta t^2 + i\omega\Delta t\right) y^n \end{aligned} \tag{4.6}$$

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

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

and we can see

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

The growth factor is

$$\begin{aligned}
\chi^2 &= \left|\left(1 - \frac{(\omega\Delta t)^2}{2} + \frac{(\omega\Delta t)^4}{24}\right) + i\left(\omega\Delta t - \frac{(\omega\Delta t)^3}{6}\right)\right|^2 \\
&= \left(1 - \frac{(\omega\Delta t)^2}{2} + \frac{(\omega\Delta t)^4}{24}\right)^2 + \left(\omega\Delta t - \frac{(\omega\Delta t)^3}{6}\right)^2 \\
&= 1 - \frac{1}{72}(\omega\Delta t)^6 + \frac{1}{576}(\omega\Delta t)^8. \qquad (4.7)
\end{aligned}$$

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.

Chapter 5

Numerical Results

In this chapter, we will present numerical solutions using the algorithms we have described in the previous section. These solutions include some benchmark tests on the spring system alone, the solution of fabric with a prescribed external velocity field, and the fluid-structure coupled solution on the evolution of parachute canopies.

5.1 Young's Modulus

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

$$E = \frac{\sigma}{\epsilon} = \frac{F}{A}\frac{l}{\Delta l} \tag{5.1}$$

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

$$E_s = \frac{F}{D}\frac{l}{\Delta l}, \tag{5.2}$$

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. 5.1. 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. In our parachute simulations, we have considered the Young's modulus given by [57], that is $2.0 \times 10^5\ lb/ft^2$ for canopy and $4.32 \times 10^6\ lb/ft^2$ for the cable. The thickness of the canopy is 0.0001 ft and the cross sectional area of the cables is 0.0001 ft^2. When converted into MKS unit, these result in $E_s = 2918.7\ N/m$ and $E_l = 1922.8\ N$. In all the simulations, we have used the spring constant 5000 for both the canopy surface and the cable. Since Hook's law is highly nonlinear on both the fabric and the cable, the excessive spring stiffness is justified.

5.2 The Oscillatory Motion of Fabric Spring Model

In Chapter 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 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. 5.2). 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. 5.3 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 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. 5.4 shows the spectra of the tangential oscillatory motion. The three plots in Fig. 5.4 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. 5.4 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.

5.3 First, Second and Fourth Order Schemes

From both the analysis in Chapter 4 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. 5.5 show the comparison between second order and fourth order schemes on Eq. (4.4). 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. (4.5). 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,

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}

Table 5.1: Comparison of numerical schemes for ODE Eq. (4.4) 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 $E = \frac{1}{2}\mu^2 x^2 + \frac{1}{2}v^2$ is the total energy and E_0 is the energy at $t = 0$.

and 40000 time steps are summarized by Table 5.1.

5.4 Wrinkling and Draping of Linear and Surface Spring Mesh

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.

In the first set of solutions, we tested both linear and surface spring system by randomly perturbing the position of each vertex (mass) point at $t = 0$. The initial velocity of each vertex point is set to zero. In both linear and surface solutions, we plotted the total spring potential energy, the total kinetic energy of the mass points, and the total mechanical energy (potential plus kinetic energy). Fig. 5.6 shows the exchange between potential energy and kinetic energy for a perturbed linear spring chord. Fig. 5.7 shows the perturbed surface mesh and the evolution of energy *vs.* time. In both cases, the fourth order Runge-Kutta method is used and the energy conservation is nearly perfect even after as many as 20,000 time steps.

In the second numerical solution, we attached a payload to the free end of the string and fix another end. The string is released at an initial height and swing under the gravitational field. There is no friction force. The potential energy in this case involves both spring potential energy and the gravitational (external) potential energy. Fig. 5.8 shows the three positions of the swinging string chord at $t = 0, 0.5, 1.0$. Again in this case, the total energy, with gravitational potential energy included, is almost perfectly conserved, as shown by Fig. 5.9.

In the third set of benchmark test, we tested a cloth modeled by the spring surface mesh evolving in an external velocity field. In the first case, we put a square cloth in a parabolic velocity field $v = v_0(x^2 + y^2)$. The cloth is driven by the external velocity, but undergoes a constrained motion due to the spring energy. Unconstrained non-fabric surface motion in the same velocity field results in cone shape geometry. Fig. 5.10 is the top view of the spring mesh which shows that at later time, the cloth developed wrinkling in response to the bending of the surface in the velocity field. The spring model does not conserve the total area exactly, but the restoring force preserves the total area approximately.

In another numerical experiment on fabric surface, we use a singular velocity field

$$v = \begin{cases} v_0 & \text{if } x^2 + y^2 < a^2 \\ 0 & \text{otherwise.} \end{cases} \tag{5.3}$$

The cloth behaves as a drapery as shown by Fig. 5.11. Even though only a few points are driven by the external velocity field, the entire fabric surface feels

the dragging force by the moving points through the spring system. Collision is handled by functions in the front tracking library. In the following section, we use the same simulation to deform the parachute canopy into the initially folded state for the inflation and drop test simulations.

5.5 Static and Rigid Body Solution

Simulations on the asymptotic steady state solution were conducted on parachute by approximating the canopy surface as an inflated rigid body thin surface. In this set of simulations, we treat the fabric surface as a Neumann boundary. The lower boundary of the computational domain is set to be a Dirichlet boundary with constant upward velocity while the upper boundary of the computational domain is a flow-through boundary. We have used periodic boundary for four sides of the computational domain. The test parachute canopy has a parabolic surface described by the equation

$$z = a\left((x-x_0)^2 + (y-y_0)^2\right) + z_0, \qquad r < \sqrt{(x-x_0)^2 + (y-y_0)^2} < R, \quad (5.4)$$

where r is the radius of the vent and R is the diameter of the canopy. Since we are interested in the asymptotic solution after a long time of relaxation, we use the velocity field which is uniform at $t = 0$ and let the Navier-Stokes equation to adjust automatically. Since the projection method eliminates the divergence of the velocity field after each time step, the velocity becomes divergence-free after a couple time steps. We continued the computation until the flow around the canopy surface and in the domain reaches a steady state (with very little

variation). We measure the drag force by computing the pressure difference between the lower and upper sides of the canopy surface and integrate over the entire canopy surface. The steady state solution can be used as a benchmark and reference to the dynamic solution. A dynamic solution can be verified by comparing with the steady state solution for a given payload and terminal velocity, especially after the parachute is fully inflated.

5.6 Solution of Parachutes

Parachute is a complex system. Except verifying certain basic benchmarks on the fluid solver, the conservation of spring energy and external energy, it is difficult to set up standard problems for a benchmark comparison. Therefore in this paper, we try to give some assessment to our numerical model and verify and validate, at least in magnitude, that the system is physically correct. Improvements may need to be added to make the parachute system more accurate and realistic. Among all the measurements and geometry, perhaps the most important physical variable of any parachute system is its terminal velocity as a function of payload. For the given geometry of a parachute canopy and payload, almost all parachute system reaches 90% of the terminal velocity in a few seconds. Without considering the buoyancy effect, a dimensional analysis gives the following force balance between gravity and the drag

$$F_{net} = Mg - \frac{1}{2}\rho V^2 A C_d, \tag{5.5}$$

where F_{net} is the net force on the system, M is the total mass of the parachute system including the mass of the system and the mass of payload, A is the area of the canopy in its inflated state, V is the center-of-mass velocity of the system, ρ is the fluid density around the canopy and C_d is a coefficient which depends on the specific type of the parachute. In steady descent, we have $F_{net} = 0$ which gives the terminal velocity

$$V_t \sim \sqrt{\frac{2Mg}{\rho A C_d}}. \tag{5.6}$$

We will show that in the system we have developed, this equation is qualitatively validated. We lack the exact value of C_d for different systems, but we show that the terminal velocities are in the reasonable range for each parachute system.

5.7 Dynamic Motion from Extended Canopy States

We applied the coupling of the spring system and fluid solver for the study of the dynamic motion of three types of parachutes. These parachutes differ in geometry and dimensions of the canopies and risers. Among these parachutes, the T-10 parachute is used by the Army as the personnel carrier. This type of parachute has a parabolic shape for the canopy with a vent at the top. The G-11 parachute is a cargo parachute with no vent. Its dimension could vary and are usually used as a multiple parachute system to deliver supporting equipments. In this paper, we will only study the single 1/3-scale G-11 parachute inflation. The cross parachute has four open side vents and can

be used as a sports parachute and small cargo carrier. These three parachutes produce different patterns of airflow around the canopy and exert different drags to the parachute system. Table 5.2 presents the specs of these three parachutes.

Type	**T-10 Personnel**	**1/3-scale G-11**	**Cross Parachute**
Shape	Parabolic	Flat Circular	Flat Cross
Diameter	10.7 m	10.1 m	1.27 m
Inflation Diameter	7.8 m	-	-
Number of Susp. Lines	30	30	20
Length of Susp. Lines	7.8 m	9.6 m	1.27 m
Assembly Weight	14 kg	-	0.227 kg
Weight Capacity	163 kg	-	-

Table 5.2: Characteristic dimensions of three types of parachutes.

We first carried out the dynamic simulations starting with the canopy surfaces in their extended states. For the T-10 parachute, it is a parabola, and for G-11 cargo parachute and the cross parachute, they are initialized as fully extended planes. The bond lengths (spring lengths) are all at their equilibrium states, therefore the initial total spring potential energy is zero. In these dynamic simulations, the canopy surface is attached by a set of string chords at the edge of the canopy surface which are then connected to the payload. For simplicity, we approximate the payload as a point mass. All string chords are also initially at the equilibrium states. We start with an uniform upward air velocity field and let it relax for 10 time steps so that the projection method will adjust the velocity field to its divergence-free state around the parachute. After that we let the interaction between the spring system and the fluid solver to start. In this set of simulations, the parachute moves in response to the gravitational force, the fluid pressure and the internal

force of the spring system. Each entity (canopy, string chords, and payload) is a set of mass points connected by the spring bond to its neighbor or neighbors. The mass of each point in the parachute entities is calculated using the total mass from its specs of the entity divided by the total number of points used to discretize the entity.

Our simulations can be summarized as follows:

(1). **The T-10 Parachute** T-10 Parachute starts with a parabolic shape. The initial extended diameter is 10.7 m. It has 30 suspension lines which are 7.8 m in length for each. A complete assembly weight is 14 kg, the maximum weight capacity is 163 kg. The reference frame of the simulation has an upward velocity 3 m/s. Although the initial state of the canopy is not realistic, the numerical solution does show some important features of the T-10 parachute. Fig. 5.12 shows T-10 personnel parachute drop test in the initial velocity field. In this figure, there is a fast air flow through the vent at the top canopy. A hole in the apex helped to vent some air and reduce the oscillations. We have simulated the payload ranging from 100 kg to 600 kg and the terminal velocity varies from approximately 3 m/s to 6 m/s.

(2). **The 1/3-scale G-11 Parachute** G-11 parachute is a cargo delivery parachute with diameter of 30m and 120 cables or string chords. Diameter of 1/3-scale G-11 parachute is 10.1 m with 30 cables. We are interested in comparing the difference between the flow in the T-10 parachute and the G-11 parachute. Fig. 5.13 illustrates the fluid flow around the parachute. For the G-11 parachute without vent, we have noticed that

the inflation process and the terminal descent are less stable. Throughout inflation process, the surrounding air becomes increasingly disturbed and the vortex flow near the top edge of the canopy is stronger in the G-11 parachute than that of the T-10 parachute. In addition, we have observed drifting of the parachute while reaching its terminal velocity state. But after the canopy is fully inflated, the surrounding air flow is stabilized in the neighborhood of terminal velocity.

(3). **The Cross Parachute** Cross Parachute is a test parachute. It is designed for wind tunnel test [53]. A diameter of center panel of canopy is 1.27 m. It has 20 suspension lines with same length as 1.27 m. A total weight of parachute is 0.227 kg. Cross parachute simulation results are shown in Fig. 5.14. We also performed inflation simulation from several folded states, and the results are shown in Fig. 5.15. This parachute will be discussed in more details in Chapter 6 and compared with the wind tunnel test.

(4). **The Test Parachute** We also initialized 2.134 m (7 ft) diameter flat circular parachute to compare with the indoor vertical drop test [20, 67]. This parachute has 16 suspension lines and payload is 5.398 kg (11.9 lbs). A set of comprehensive indoor vertical parachute drop tests were carried out to collect data for validation and the study of the physics of parachute dynamics. In these experiments, parachutes with diameter of 1.067 m (3.5 ft), 2.134 m (7.0 ft), and 2.743 m (9.0 ft) were vertically dropped in an indoor controlled environment to obtain the measurements. Because

the reference frame in our simulation has an upward ambient velocity, the numerical solution cannot be compared with experimental data for the beginning time period of the simulations. But the terminal descent velocity and breathing frequency in the parachute simulations are in good agreement with vertical parachute test result as shown in Fig. 5.16. The comparison will be discussed in Chapter 6.

5.8 Inflation from Folded Canopy States

For canopy inflation test, we deform the parachute geometry into several folded states from fully extended state. For this pre-simulation, the singular velocity field Eq. (5.3) is used. The pressures both inside and outside of the canopy are very difficult to calculate in the folded state. The coating algorithm described in Sec. 2.11 helps very little in getting the accurate pressure difference at points which are in contact with others. The current approximation is to use the average pressure from the points on a given side interpolatable from the fluid as the pressure of those points whose pressure cannot be determined. The pressure difference becomes more and more accurate as the canopy is opened. We have experimented inflation from different stages of the folded state. Even the simulations start from different starting states, the final velocities are converged to a range around the terminal velocity after several seconds. Fig. 5.17, and Fig. 5.18 show the several initial states of G-11 and Cross parachutes respectively. Fig. 5.15 shows the simulation of the cross parachute inflation process.

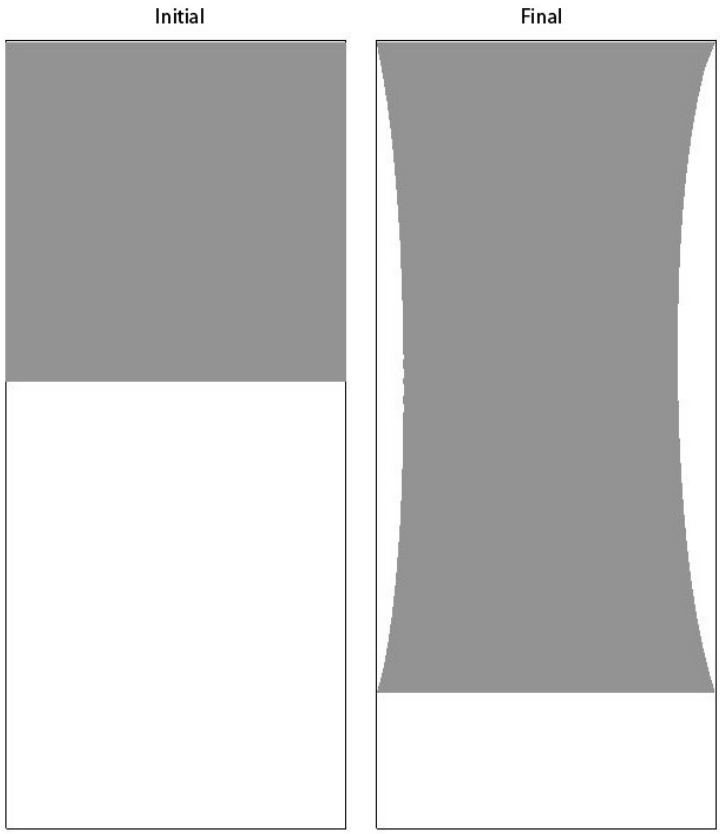

Figure 5.1: Numerical simulation of Young's modulus on surface. A force is uniformly applied to the bottom boundary and the surface is pulled downward. The middle part of the surface is deformed inward resulting in a concavity of the two vertical sides. The resulting Young's surface modulus E_s is about 1.1 times of the spring constant k_s.

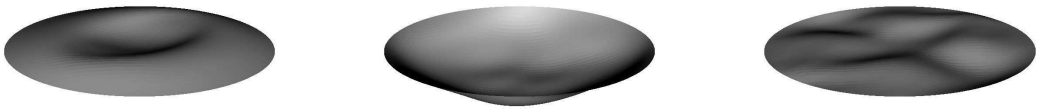

Figure 5.2: The plots in this figure show a fabric surface vibrating motion. The surface is fixed at the circular boundary. An initial perturbation is applied to the fabric surface. We use this simulation to study the motion of mass point tangential to the surface and normal to the surface.

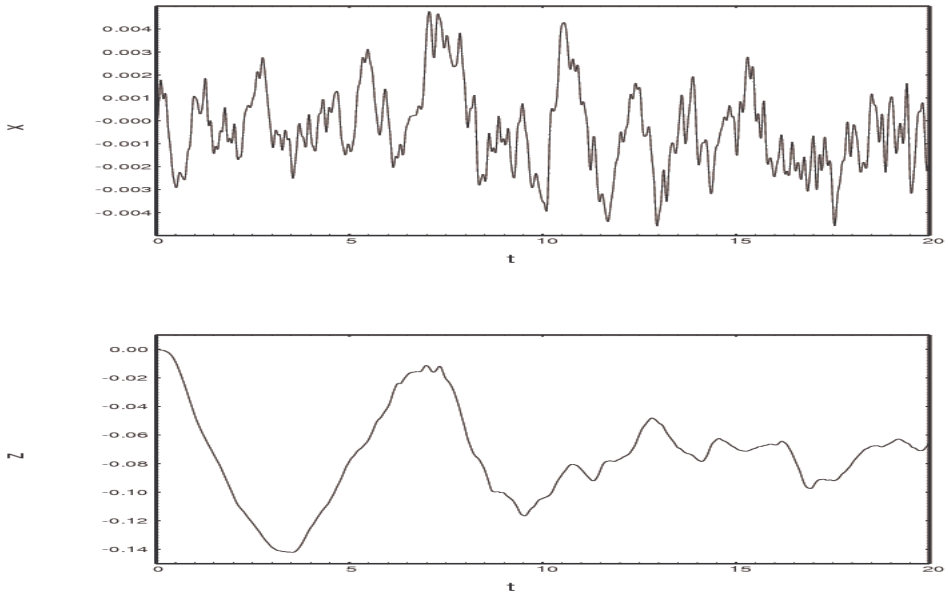

Figure 5.3: Relative displacement of a sample point on the fabric surface as a function of time. The top is the displacement almost tangential to the surface (the x-coordinate), the bottom is the displacement almost normal to the fabric surface. The tangential motion is oscillatory whose frequency is 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.

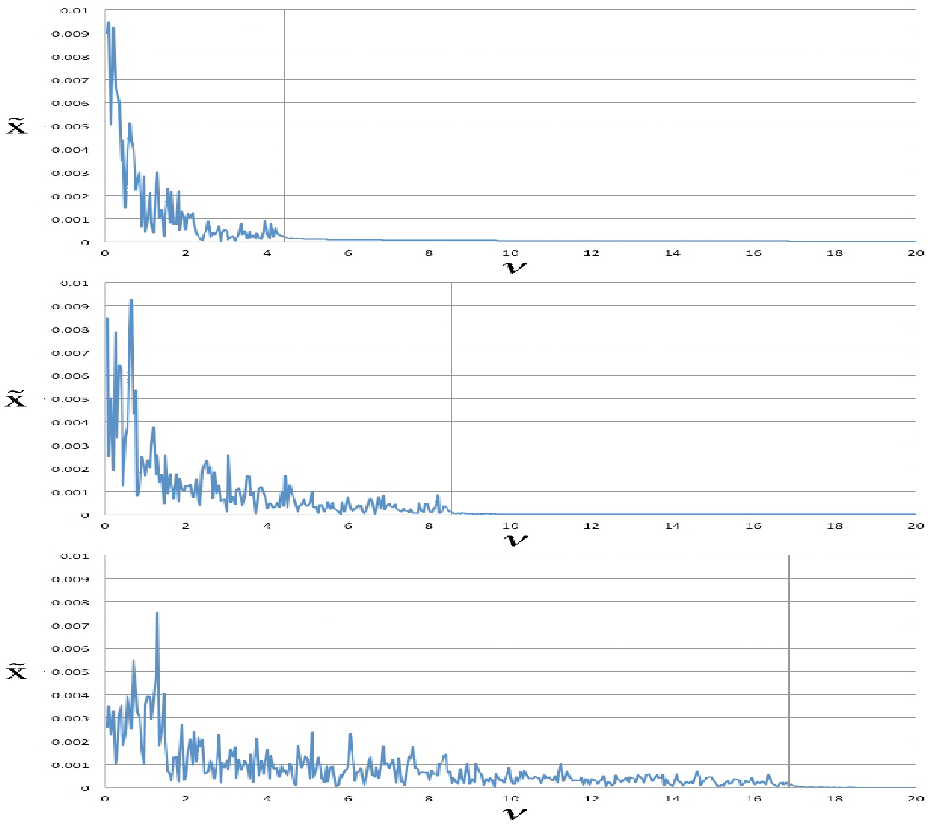

Figure 5.4: The three plots in the figure are the Fourier transformation of the tangential displacement function. The vertical line in each plot is the cut-off frequency $\omega_c = \sqrt{Mk/m}$ with $M = 7$. The spring constant in all three cases is 1000, and the point masses for three cases from top to bottom are $10, 2.5, 0.625$ respectively. These plots prove that the tangential oscillation frequency is bounded by $\omega_c = \sqrt{Mk/m}$.

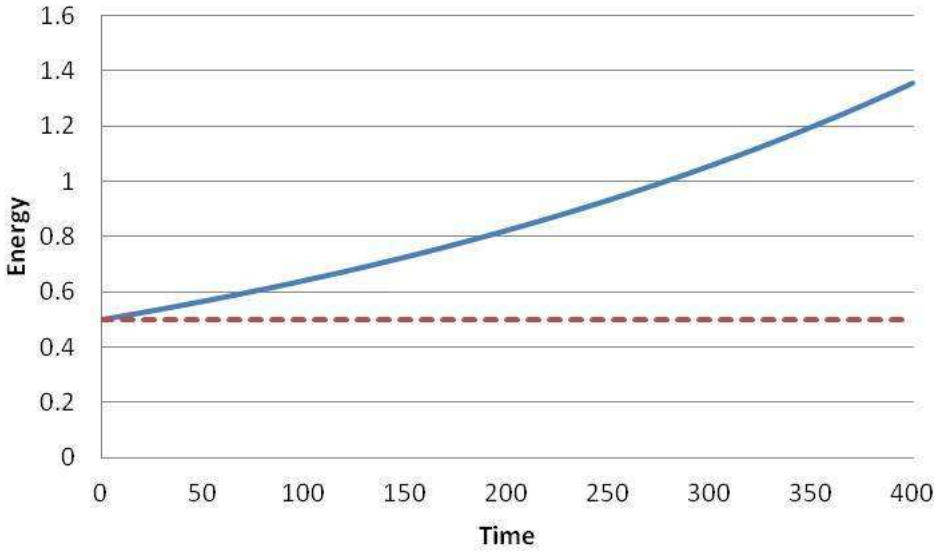

Figure 5.5: Comparison of the second and fourth order schemes for Eq. (4.4) with $\omega\Delta t = 0.1$ over 40,000 time steps. The plot shows the comparison of the total energy. The numerical results show that the second order predictor–corrector method added 1.7 time of the total energy to the initial value after 40,000 time steps while the fourth order Runge-Kutta method only changed 0.05 percent.

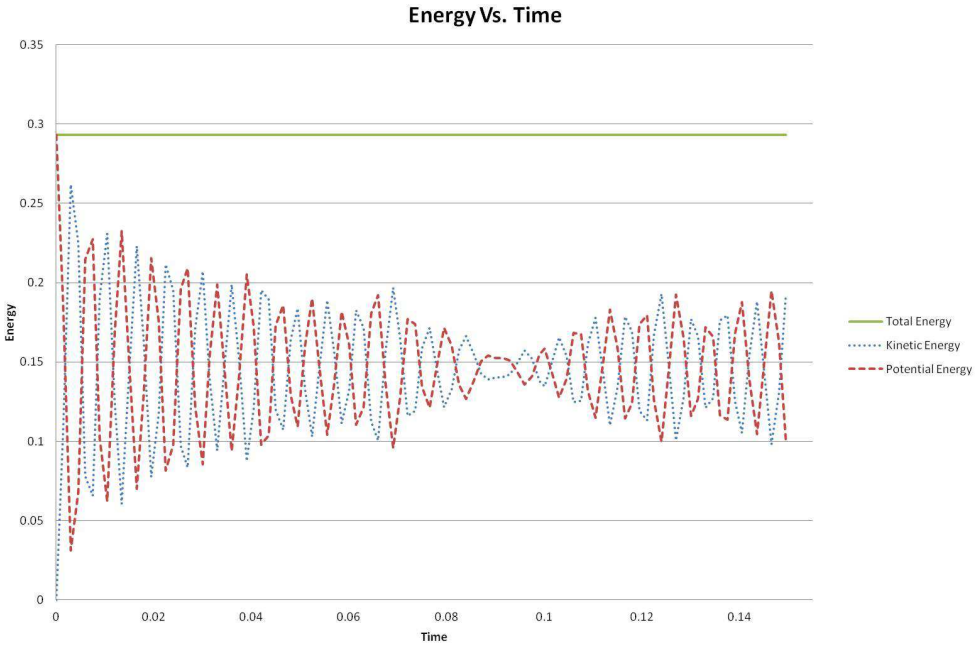

Figure 5.6: The energy evolution of a perturbed string chord without external driving force. This plot shows the exchange between spring potential energy and kinetic energy. The total energy is conserved within 0.001 percent after 20,000 time steps.

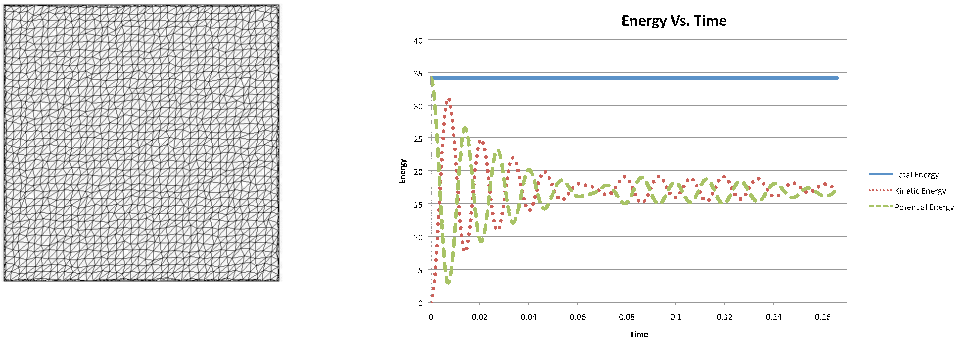

Figure 5.7: A surface spring mesh is perturbed at $t = 0$ (left). The system exchange between spring potential energy and kinetic energy. Using the fourth order Runge-Kutta method, the total energy is almost perfectly conserved after up to $20,000$ time steps.

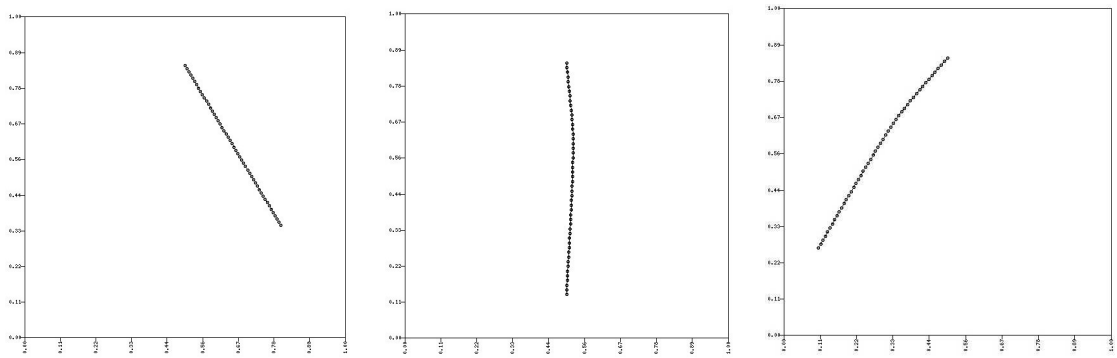

Figure 5.8: The spring model applied to a string chord with 50 mass points. In this simulation, the spring constant is set to $k = 5000$, and mass of each point is $m = 0.01$, together with gravity $g = -9.8$ and payload $w = 1$.

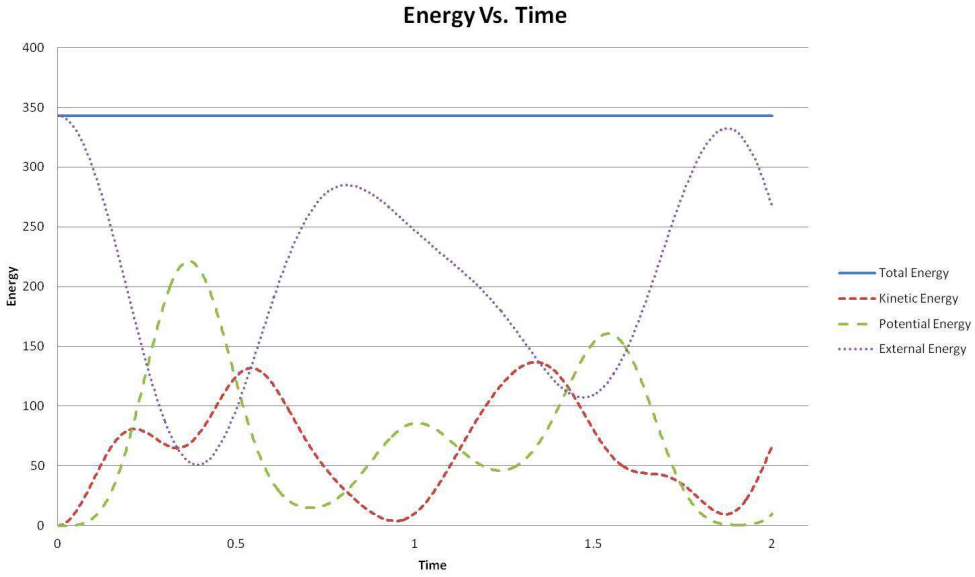

Figure 5.9: The energy of a string chord swing. This plot shows the exchange between spring potential energy and kinetic energy with external (gravitational) potential energy. The total energy is conserved.

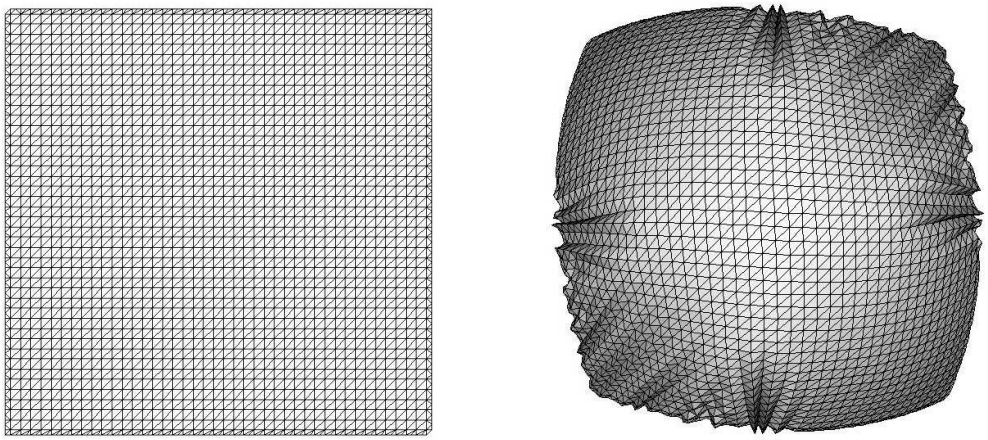

Figure 5.10: Top view of the evolution of a square cloth driven by a parabolic velocity field $v = v_0(x^2+y^2)$. While the cloth responds to the external velocity field, the spring system automatically adjusts the total internal energy toward minimum state resulting in the wrinkles at the edges of the cloth.

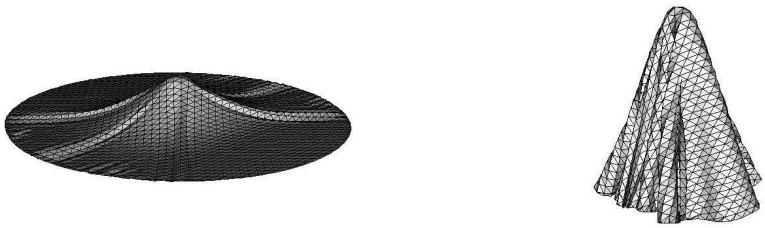

Figure 5.11: Simulation of a cloth with a pulling velocity at the center. The fabric constraint automatically adjusts the parts of the cloth. The spring model of the fabric gives a realistic motion of the cloth.

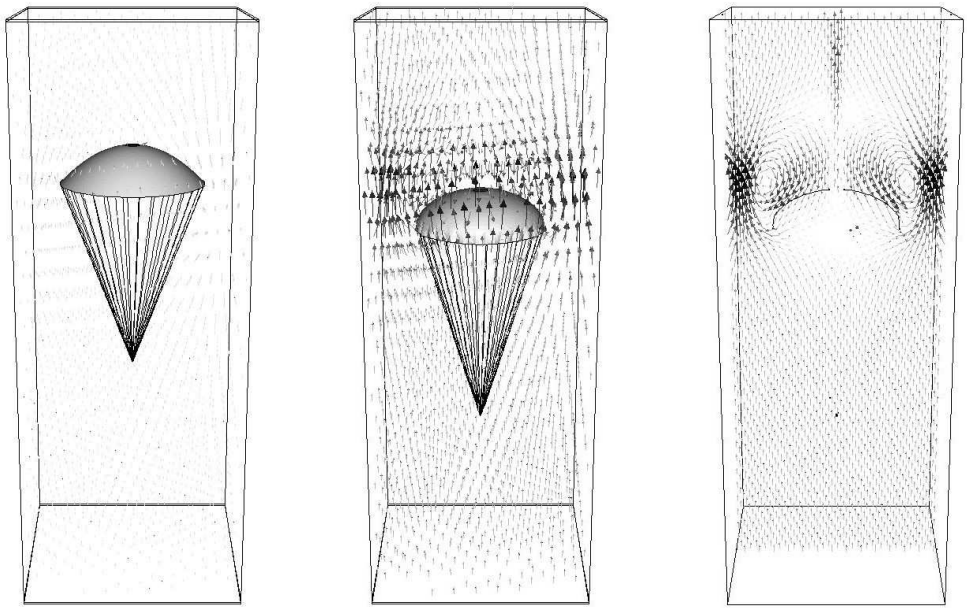

Figure 5.12: T-10 personnel parachute is a parabolic parachute with a vent on top of the canopy. A diameter is $10.7m$, it has 30 suspension lines which are $7.8m$ for each. A complete assembly weight is $14kg$, the maximum weight capacity is $163kg$.

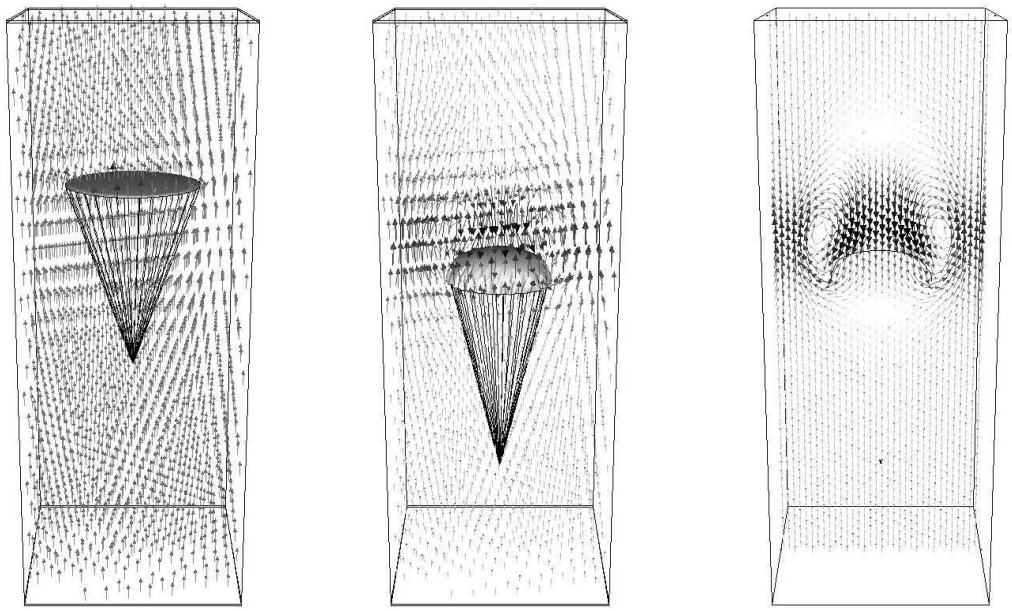

Figure 5.13: This is a simulation on a 1/3-scale G-11 parachute. G-11 parachute is a flat-circular shape parachute. This simulation is for purpose of comparing with T-10 parachute using the same diameter and the same number of cables, but no vent at top. The reference frame has an upward fluid velocity of 3 m/s and the payload of 150 kg.

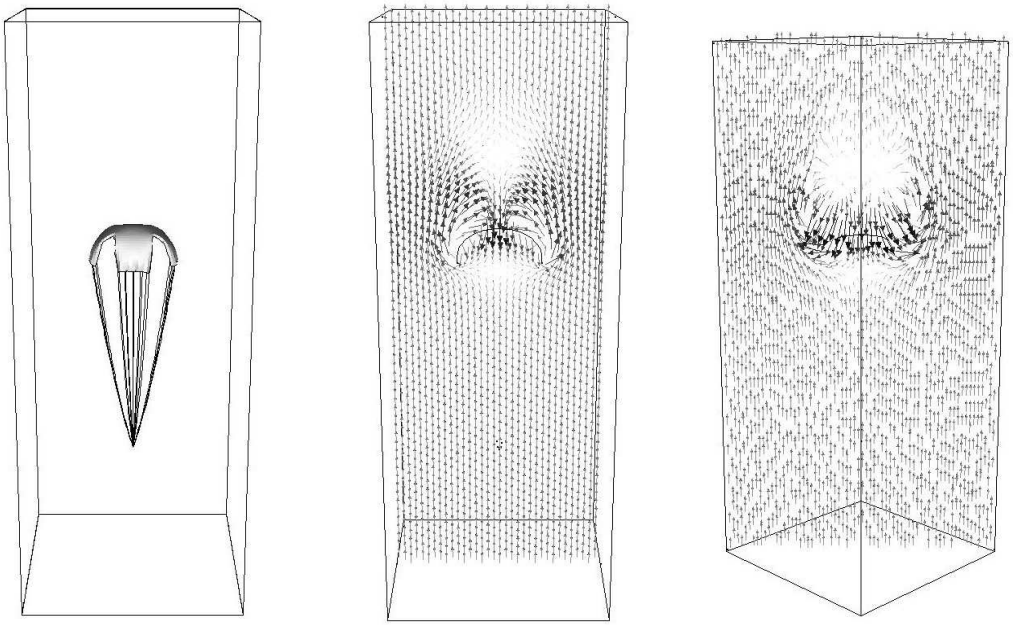

Figure 5.14: Cross parachute is used for wind tunnel experiment. The initial shape is flat cross, a diameter is 1.27 m, it has 20 suspension lines which are 1.27 m each.

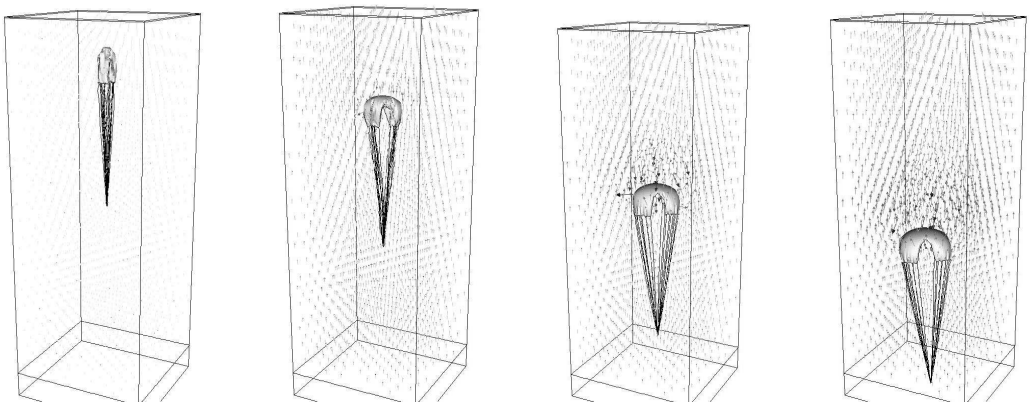

Figure 5.15: Simulation of cross parachute unfolding and inflation. The starting state of the parachute is deformed from the fully extended state through the singular velocity field Eq. (5.3). The parachute has a small horizontal drift because the folded state is not perfect symmetric.

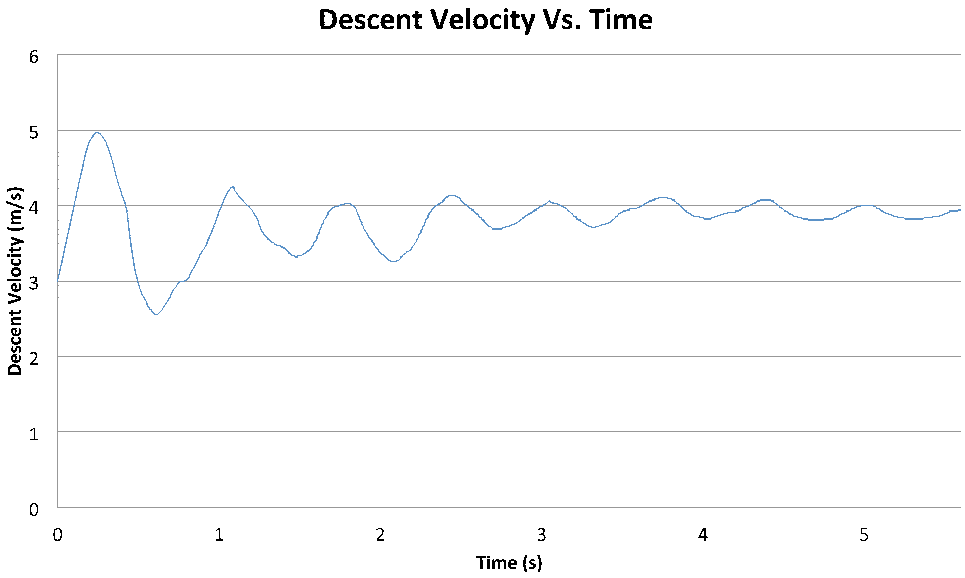

Figure 5.16: The numerical result of descending velocity *vs.* time for a parachute with payload of 5.398 kg (11.9 lbs) and canopy diameter 2.134 m (7 ft). For the simulation, there is an upward velocity; therefore the graph starts from the ambient velocity. However, terminal velocity of parachute reaches to the value at about $3.9 \pm 0.05 m/s$. The experimental steady descent speed approaches to 4.27 m/s (14 ft/s), and the terminal velocity of simulation is $4.27 \pm 0.05 m/s$. The difference is believed due to the lack of porosity in the current model.

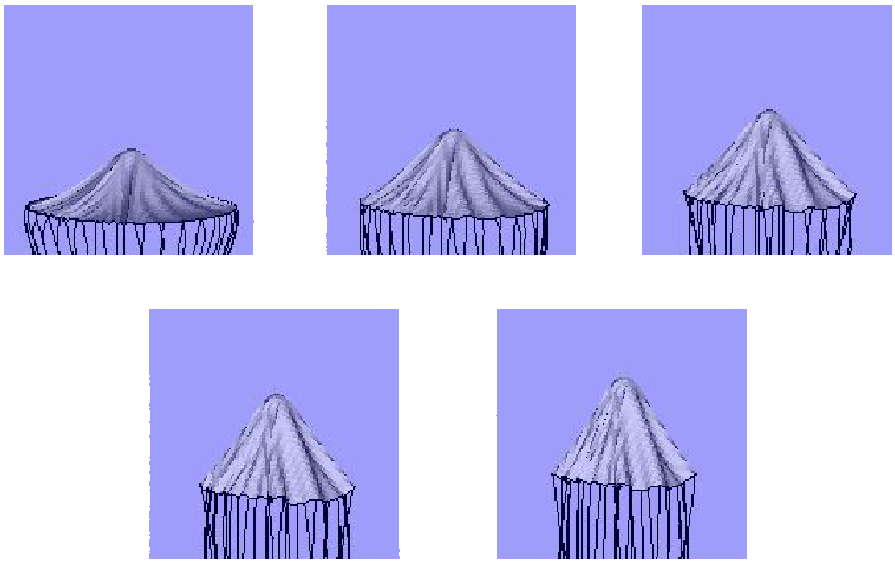

Figure 5.17: Initial folded states of 1/3-scale G-11 parachute. For this pre-simulation, the singular velocity field Eq. (5.3) is used from fully extended canopy.

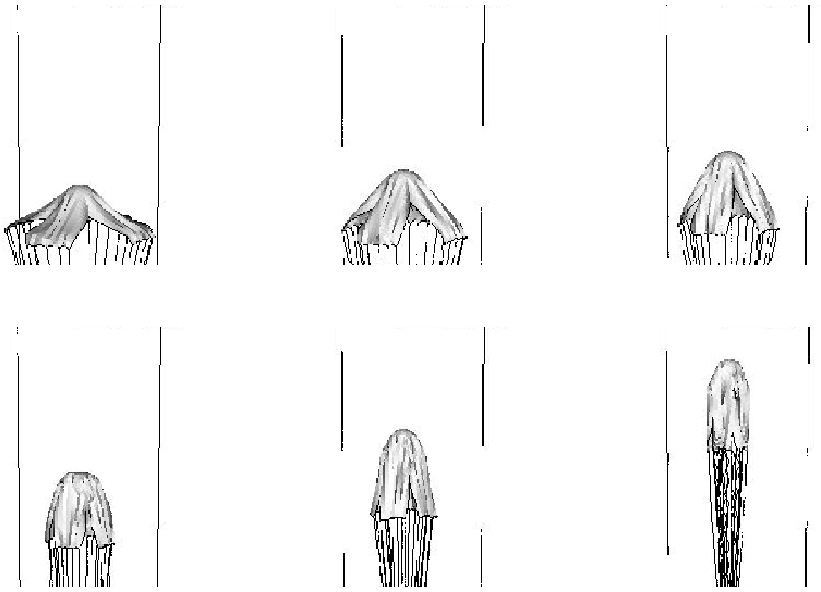

Figure 5.18: Initial folded states of Cross parachute. For this pre-simulation, the singular velocity field Eq. (5.3) is used from fully extended caonpy.

Chapter 6

Validation by Comparison

There are several experiments for parachute drop test. In this paper, we compare two data sets to verify the numerical results: wind tunnel test with cross parachute [53] and indoor vertical parachute test, conducted at the Space Power Facility of NASA Glenn Research Center Plum Brook Station [20, 67]. The dimension of the cross parachute in this simulation is the same as the parachute in the wind tunnel test, but there is no reinforcements in the canopy along the seams and outer edges. Without reinforcements we could get a good agreement with their experiment result. Fig. 6.1 shows fully inflated canopy shape from *FronTier*.

For comparison with the indoor vertical parachute test, we initialized the parachute with flat circular canopy of 2.134 m (7 ft) nominal diameter. We carried out pre-step running to deform the parachute canopy and then used the resulting geometry as the initial state for the drop test simulation. The initial skirt diameter makes different time graph for full inflation. In this test, there is a breathing motion or the over-inflation of the canopy. It is

shown that the canopy progresses into an over-inflated shape, almost flat, and then contract and over-inflates again. The breathing motion is an oscillatory motion. The canopy appeared to expel the excess air by means of the breathing. In experiment, the breathing motion was also caused by the constraint on the parachute, imposed by the guide wire. The breathing motion in the simulation is smaller because there is no vertical motion restriction such as guide wire. The terminal velocity of canopy has a good agreement with the vertical parachute test results as shown in Fig. 5.16. The experimental data shows that the descent speed rises rapidly to a peak. It slows down while the parachute inflates and then slowly approaches a steady descent speed at 4.27 m/s (14 ft/s). Even though the numerical solution cannot be compared with the experimental data during the initial period of time in the simulation, we have observed that it reaches the terminal velocity at about 3.9 m/s. The difference between the terminal velocity in numerical simulation and the experimental data is thought due to the lack of porosity of the canopy in the current numerical model.

We also simulated the 1/3-scale G-11 parachute with different canopy folding level, they all converge to the terminal speed of 3 m/s in several seconds. Fig. 6.2 shows the descending velocity as a function of time in these simulations.

Parachute breathing is an oscillatory motion caused by the interaction between the fluid force and the canopy at the skirt of the parachute. The large difference between upper and lower side pressure causes vibration in the projected drag area of the canopy and thus the oscillation in descent speed. Nu-

merical study of breathing frequency is important to understand the stability of the parachute. In our simulation, we initialized 2.124 m (7 ft) flat-circular parachute to compare with experimental data by Tutt *et al.*[67]. Fig. 5.16 shows that the breathing frequency is approximately 1.6 - 2.0 Hz (0.5 s to 0.6 s period). This is in an acceptable range with experimental frequency which is 2.0 Hz (0.5 s period). If we use parachute with larger size or lower Reynold number, the breathing period will increase. For example, the average period of breathing for the 10.7 m T-10 parachute is 2.3 s in experiment [39], while it is about 2 s in our simulation. Fig. 6.3 shows the shape change of the C-9 parachute canopy during the breathing motion.

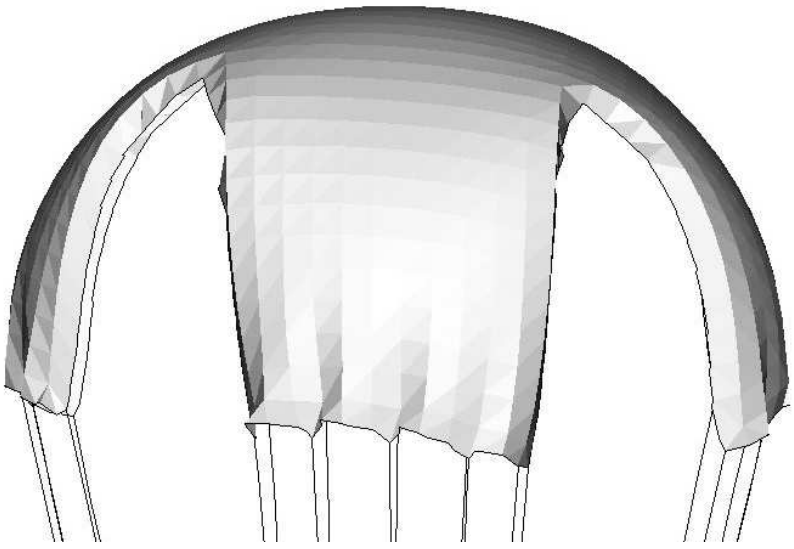

Figure 6.1: Fully inflated canopy of cross parachute from *FronTier*

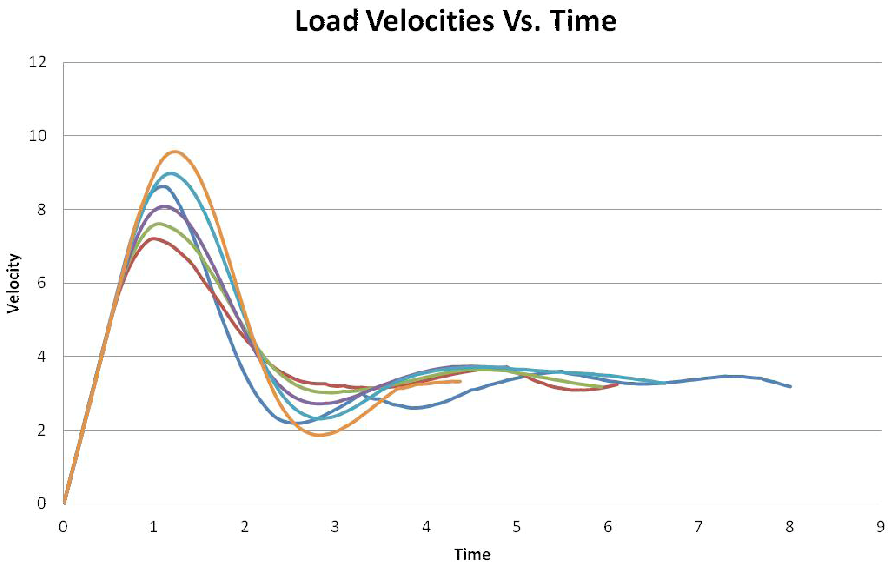

Figure 6.2: Descending velocities of 1/3-scale G-11 parachute with several initial states. This figure shows descending velocities converge to safe landing speed in several seconds, even they started from different initial folded states.

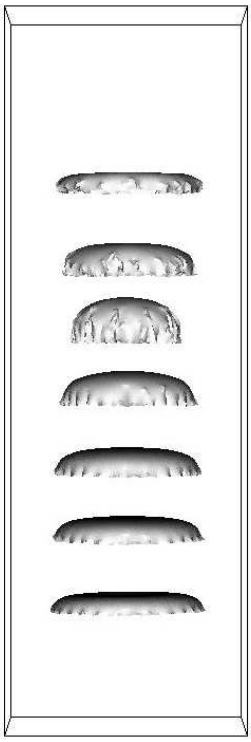

Figure 6.3: This figure shows the breathing motion of C-9 parachute canopy. C-9 parachute is a flat-circular personal parachute with radius of 8.53 m (28 ft). The breathing period is approximately 2 s.

Chapter 7

Conclusion

We use the front tracking data structure and functionalities to model the dynamic motion of fabric material and parachute inflation. 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 exists an upper bound $|\mu| \leq \sqrt{2Mk/m}$, 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.

In our method, both the parachute canopy and the risers are modeled by the spring system. The fabric responds to the velocity field with reasonable wrinkling and draping. For the simulation of the dynamic motion of parachute in air, we coupled the solution of the spring system with the Navier-Stokes solver. We have introduced index coating algorithm for sharp interpolation of air pressure on the two sides of the canopy. This algorithm gives accurate pressure difference when the canopy is fully opened, but may need further improvement in the folded state of the canopy. We studied the T-10 personnel parachute, the G-11 cargo parachute, the cross parachute, and the 2.134 m

(7 ft) flat-circular parachute. The cross parachute and the 2.134 m flat-circular parachute simulations are compared with the available experimental data. These comparisons showed similar geometric shape and good agreement on terminal velocity and breathing frequency after inflation.

We have used the realistic payload in all simulations. We found that the spring model gives excellent and physically reasonable response of fabric to the stretching and tension by air pressure and the spread the force of payload to the fabric spring system. Our coupling algorithm, namely the "Impulse Method", separates the impacts of the internal and the external forces and eliminates unphysical damping.

More careful treatment on the calculation of pressure at the canopy surface is needed in the folded state. In the current model, the parachute canopy is modeled as a fabric-only structure without the radial reinforcement cables, and without taking into account the fabric porosity. Other amendments such as the inclusion of gore and secondary risers are needed for more accurate and realistic simulations of the parachute system. The inclusion of gore or the radial reinforcement structure requires changes and modifications of the data structure and functions in the front tracking library. As an internal curve, the gore is viewed by two adjacent surface pieces with different opposite orientation. We are revising the front tracking library to address this issue. For the cargo parachute, we also need to implement the multi-parachute interaction. We plan to discuss these new issues in the next paper. Our terminal descending velocity is slightly lower than that is observed in experiment; we believe this is due to the lack of porosity in the current model. Our future study will

include finite porosity at the canopy surface.

Bibliography

[1] R. M. Aileni, D. Farima, and M. Ciocoiu. Simulating cloth in realistic way using vertices model based. *7th International Conference-TEXSCI*, 2010.

[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] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Annual Conference on Computer Graphics*, pages 43–54, 1998.

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

[8] J. B. Bell, P. Colella, and H. M. Glaz. An efficient sceond-order projection method for viscous incompressible flow. *Proceedings of the Tenth AIAA Computational Fluid Dynamics Conference*, AIAA:360, 1991.

[9] 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.

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

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

[12] 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.

[13] 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.

[14] 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.

[15] D. Brown, R. Cortez, and M. Minion. Accurate projection method for the incompressible Navier Stokes equations. *J. Comp. Phys.*, 168:464–499, 2001.

[16] 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.

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

[18] A. J. Chorin. Numerical solution of the Navier Stokes equations. *Math. Comp*, 22:745–762, 1968.

[19] A. J. Chorin. On the convergence of discrete approximations to the Navier-Stokes equations. *Math. Comp*, 23:341, 1969.

[20] K. J. Desabrais, C. K. Lee, J. Buckley, and T. W. Jones. Experimental parachute validation research program and status report on indoor drop tests. *19th AIAA Aerodynamic Decelerator Systems Technology conference and Seminar*, 2007.

[21] J. Du, B. Fix, J. Glimm, X. Jiao, X. Li, Y. Li, and L. Wu. A simple package for front tracking. *J. Comput. Phys.*, 213:613–628, 2006.

[22] 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.

[23] S. Dutta, E. George, J. Glimm, J. Grove, H. Jin, T. Lee, X. Li, D. H. Sharp, K. Ye, Y. Yu, Y. Zhang, and M. Zhao. Shock wave interactions in spherical and perturbed spherical geometries. *Nonlinear Analysis*, 63:644–652, 2005. University at Stony Brook preprint number SB-AMS-04-09 and LANL report No. LA-UR-04-2989.

[24] S. Dutta, E. George, J. Glimm, X. L. Li, A. Marchese, Z. L. Xu, Y. M. Zhang, J. W. Grove, and D. H. Sharp. Numerical methods for the determination of mixing. *Laser and Particle Beams*, 21:437–442, 2003. LANL report No. LA-UR-02-1996.

[25] 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.

[26] 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.

[27] C. L. Gardner, J. Glimm, J. W. Grove, O. McBryan, R. Menikoff, D. H. Sharp, and Q. Zhang. A study of chaos and mixing in Rayleigh-Taylor and Richtmyer-Meshkov unstable interfaces. In M. Duong-van and B. Nichols, editors, *Proceedings of the International Conference on 'The Physics of Chaos and Systems Far from Equilibrium' (CHAOS' 87), Monterey, CA, USA, Jan. 11–14, 1987*. 1988. Special Issue of Nuclear Physics B (proceedings supplements section).

[28] 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.

[29] E. George, J. Glimm, X. L. Li, A. Marchese, and Z. L. Xu. A comparison of experimental, theoretical, and numerical simulation Rayleigh-Taylor mixing rates. *Proc. National Academy of Sci.*, 99:2587–2592, 2002.

[30] J. Glimm, X.-L. Li, R. Menikoff, D. H. Sharp, and Q. Zhang. A numerical study of bubble interactions in Rayleigh-Taylor instability for compressible fluids. *Phys. Fluids A*, 2(11):2046–2054, 1990.

[31] J. Glimm, X. L. Li, W. Oh, A. Marchese, M.-N. Kim, R. Samulyak, and C. Tzanos. Jet breakup and spray formation in a diesel engine. In *Proceedings of the Second MIT Conference on Computational Fluid and Solid Mechanics*. Cambridge, MA, 2003. SUNY Stony Brook preprint No. susb-ams-02-20.

[32] J. Glimm, O. McBryan, R. Menikoff, and D. Sharp. Front tracking applied to Rayleigh-Taylor instability. *SIAM J. Sci. Stat. Comput.*, 7:230–251, 1986.

[33] James Glimm, M.-N. Kim, X.-L. Li, R. Samulyak, and Z.-L. Xu. Jet simulation in a diesel engine. In *Computational Fluid and Solid Mechanics 2005*. Elsevier Science, 2005.

[34] Katuhiko Goda. A multistep technique with implicit difference schemes for calculating two- or three-dimensional cavity flows. *J. Comput. Phys.*, 30:76–95, 1979.

[35] S.-W. Hsiao and R.-Q. Chen. A method of drawing cloth patterns with fabric behavior. *5th WSEAS International Conference on Applied Computer Science*, pages 635–640, 2006.

[36] 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.

[37] F. Ji, R. Li, and Y. Qiu. Simulate the dynamic draping behavior of woven and knitted fabrics. *J. Industrial Textiles*, 35:201–214, 2006.

[38] G. Jiang and C.-W. Shu. Efficient implementation of weighted ENO schemes. *J. Comput. Phys.*, 126:202–228, 1996.

[39] Hamid Johari and Kenneth J. Desabrais. Vortex shedding in the near wake of a parachute canopy. *J. Fluid Mech.*, 536:185–207, 2005.

[40] Vinay Kalro and Tayfun E. Tezduyar. A parallel 3D computational method for fluid-structure interactions in parachute systems. *Computer Methods in Applied Mechanics and Engineering*, 190:321–332, 2000.

[41] K. Karagiozis, R. Kamakoti, F. Cirak, and C. Pantano. A computational study of supersonic disk-gap-band parachutes using large-eddy simulation coupled to a structural membrane. *Journal of Fluids and Structures*, 27(2):175–192, 2011.

[42] S. Kawabata. The standardization and analysis of hand evaluation. *J. Text. Mach. Soc. Japan*, 1980.

[43] J. Kim and P. Moin. Application of a fractional-step method to incompressible Navier-Stokes equations. *J. Comput. Phys.*, 59:308, 1985.

[44] Y. Kim and C. S. Peskin. 2-D parachute simulation by the immersed boundary method. *SIAM J. Sci. Comput.*, 28:2294–2312, 2006.

[45] Y. Kim and C. S. Peskin. 3-D parachute simulation by the immersed boundary method. *Comput. Fluids*, 38:1080–1090, 2009.

[46] H. Lim, Y. Yu, H. Jin, D. Kim, H. Lee, J. Glimm, X.-L. Li, and D. H. Sharp. Multi scale models for fluid mixing. *Compu. Methods Appl. Mech. Engrg.*, 197:3435–3444, 2008. Stony Brook University Preprint SUNYSB-AMS-07-05.

[47] 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.

[48] J. W. Purvis. Prediction of line sail during lines-first deployment. *AIAA 21st Aerospace Sciences Meeting*, 1983.

[49] J. W. Purvis. Numerical prediction of deployment, initial fill, and inflation of parachute canopies. *8th AIAA Aerodynamic Decelerator and Balloon Technology Conference*, 1984.

[50] R. Samulyak, T. Lu, and P. Parks. A hydromagnetic simulation of pellet ablation in electrostatic approximation. *Nuclear Fusion*, 47:103–118, 2007.

[51] 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.

[52] K. Stein, R. Benney, V. Kalro, T. E. Tezduyar, J. Leonard, and M. Accorsi. Parachute fluid-structure interactions: 3-D computation. *Comput. Methods Appl. Mech. Engrg*, 190:373–386, 2000.

[53] K. Stein, R. Benney, T. Tezduyar, and J. Potvin. Fluid-structure interactions of a cross parachute: numerical simulation. *Comput. Methods Appl. Mech. Engrg*, 191(6-7):673–687, 2001.

[54] K. Stein, T. Tezduyar, V. Kumar, S. Sathe, R. Benney, E. Thornburg, C. Kyle, and T. Nonoshita. Aerodynamic interactions between parachute canopies. *J. Appl. Mech.*, 70:50–57, 2003.

[55] K. R. Stein, R. J. Benney, V. Kalro, A. A. Johnson, and T. E. Tezduyar. Parallel computation of parachute fluid-structure interactions. *14th Aerodynamic Decelerator Systems Technology Conference*, 1997.

[56] K. R. Stein, R. J. Benney, E. C. Steeves, Development U.S. Army Natick Research, and Engineering Center. *A computational model that couples aerodynamic and structural dynamic behavior of parachutes during the opening process.* Technical report (U.S. Army Natick Laboratories). United States Army Natick Research, Development and Engineering Center, Aero-Mechanical Engineering Directorate, 1993.

[57] K. R. Stein, R. J. Benney, T. E. Tezduyar, J. W. Leonard, and M. L. Accorsi. Fluid-structure interactions of a round parachute: Modeling and simulation techniques. *J. Aircraft*, 38:800–808, 2001.

[58] J. H. Strickland, V. L. Porter, G. F. Homicz, and A. A. Gossler. Fluid-structure coupling for lightweight flexible bodies. *17th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, 2003.

[59] K. Takizawa, C. Moorman, S. Wright, T. Spielman, and T. E. Tezduyar. Fluid-structure interaction modeling and performance analysis of the orion spacecraft parachutes. *International Journal for Numerical Methods in Fluids*, 65:271–285, 2011.

[60] Kenji Takizawa, Timothy Spielman, and Tayfun E. Tezduyar. Space-time FSI modeling and dynamical analysis of spacecraft parachutes and parachute clusters. *Computational Mechanics*, 48:345–364, 2011.

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

[62] 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.

[63] 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.

[64] T. E. Tezduyar, S. Sathe, R. Keedy, and K. Stein. Space-time finite element techniques for computation of fluid-structure interactions. *Computer Methods in Applied Mechanics and Engineering*, 195:2002–2027, 2006.

[65] T. E. Tezduyar, S. Sathe, M. Schwaab, J. Pausewang, J. Christopher, and J. Crabtree. Fluid-structure interaction modeling of ringsail parachutes. *Computational Mechanics*, 43:133–142, 2008.

[66] T. E. Tezduyar, K. Takizawa, C. Moorman, S. Wright, and J. Christopher. Space-time finite element computation of complex fluid-structure interactions. *International Journal for Numerical Methods in Fluids*, 64:1201–1218, 2010.

[67] B. Tutt, S. Roland, R. D. Charles, and G. Noetscher. Finite mass simulation techniques in LS-DYNA. *21st AIAA Aerodynamic Decelerator Systems Technology conference and Seminar*, 2011.

[68] B. A. Tutt and A. P. Taylor. The use of LS-DYNA to simulate the inflation of a parachute canopy. *18st AIAA Aerodynamic Decelerator Systems Technology conference and Seminar*, 2005.

[69] J Van Kan. A second-order accurate pressure-correction scheme for viscous incompressible flow. *SIAM Journal on Scientific and Statistical Computing*, 7(3):870–891, 1986.

[70] 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.

[71] Li Yu and Xiao Ming. Study on transient aerodynamic characteristics of parachute opening process. *Acta Mechanica Sinica*, 23:627–633, 2007.