# Real-time Tissue Deformation and Image Rendering for Computer Simulation

Qian Yin
College of Information Science and Technology
Beijing Normal University
Beijing 100875, P.R. China
yinqian@bnu.edu.cn

Zhiyong Yuan
School of Computer Science
Wuhan University
Wuhan, Hubei 430072, P.R. China
zhiyongyuan@whu.edu.cn

Zukuan Wei
School of Computer Science and Engineering
University of Electronic Science and Technology of China
Chengdu, P.R.China 610054
anlexwee@uestc.edu.cn

Xin Zheng
College of Information Science and Technology
Beijing Normal University
Beijing 100875, P.R. China
zhengxin@bnu.edu.cn (Corresponding Author)

***Abstract***—Application of Virtual Reality (VR) has attracted many researchers in computer simulation fields. In this paper, an improved mass-spring model is proposed aiming at realistic simulation of tissue deformation. Firstly, virtual springs are proposed to represent the surface. Secondly, Verlet integration is applied to calculate the position of mass points during deformation without explicit computation of their velocities. Finally, a bilinear interpolation method is employed to generate a smooth mesh to render the deformed surface. The proposed method has been implemented using OpenGL which generates realistic deformation images in real-time.

***Keywords***—**Deformation, Computer Simulation, Virtual Reality, Improved Mass-Spring Model**

## I. INTRODUCTION

Computer simulators based on virtual reality have emerged in many fields. As an important component in a computer simulator, deformation has attracted much research attention. In order to design a suitable system for practical application, the simulated deformation must be implemented in real-time and have a realistic visual effect. How to build a simulator satisfying both essential requirements is still an active area of research. Most of the available systems utilize the isosurface based rendering techniques, which model surface as 3D level sets. Although these techniques can be rapidly implemented, they are often criticized for lacking of necessary visual details in some demanding systems, such as surgical simulator [1,2,3]. In this paper, a different method was proposed. The surface was modeled as a mesh of mass points, each of which is associated with a set of springs.

In 1980s, a mass-spring model was studied to simulate cloth deformation [4, 5, 6, 7, 8]. The algorithms based on this model were usually featured with fast computation and realistic visual effects. However, the deformation in some strict visual simulator, such as surgical simulator, bears a few differences from the cloth deformation: 1) The cloth can be considered as a thin layer. Therefore, a 2D mass-spring model usually suffices the modeling. Soft tissue, on the other hand, has a volume, which can not be well modeled by a 2D mesh. 2) Soft tissue has very different physical properties from the cloth made of a certain fabric. 3) The external forces exerted on the soft tissue and those exerted on clothes are quite different.

Therefore, we propose the following model to extend the existing model for soft tissue deformation simulation and implemented by the experiments. Firstly, we use a spring that is perpendicular to the surface at each mass point to model the internal force below the surface, which compensates the weakness of the 2D mass-spring model. Secondly, we employ Verlet integration to calculate the new positions of the mass points in the process of deformation [9]. Our method avoids the explicit computation of the velocities of the mass points, therefore reduces the computational complexity.

## II. METHODOLOGY

### A. *The conventional mass-spring model*

The conventional mass-spring model is a mesh containing m×n points [10]. Each point is assigned a mass and linked to its neighbors by massless springs with a nonzero length.

There are three types of springs: “structural springs” linking (i,j) to (i+1,j) and (i,j) to (i,j+1); .shear springs linking (i,j) to (i+1,j+1) and (i+1,j) to (i,j+1); and “flexion springs” linking (i, j) to (i+2, j+1) and (i, j) to (i, j+2).

### B. *Improved 3D mass-spring model*

We employed structural springs, shear springs and virtual springs in the improved mass-spring model. The virtual spring reacts to the external force as shown in Figure 1. The initial length of this spring is zero. Here, the mass point on which the external force is exerted is called control point, others are called non-control points.

978-1-4244-1674-5/08 /$25.00 ©2008 IEEE

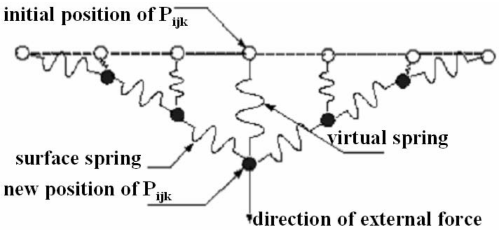
initial position of $P_{ijk}$
surface spring
virtual spring
new position of $P_{ijk}$
direction of external force

Figure 1. The principle of virtual springs

### C. Force Model

There are two types of forces, external force and internal force. In our application, we consider two external forces applied to a mass point: the surgical force and the damping force, denoted by $\vec{F}_{surgical}$ and $\vec{F}_{damp}$, respectively. For the internal forces, we consider the force generated by the structural, shear and flexion springs, all denoted by $\vec{F}_{elastic}$, and the force of the virtual spring, denoted by $\vec{F}_{virtual}$.

For a point $P_{ijk}$, we have:

$$\vec{F}_{ijk} = \vec{F}_{surgical} + \vec{F}_{damp} + \vec{F}_{elastic} + \vec{F}_{virtual} = m \cdot \vec{a}_{ijk} \quad (1)$$

where $m$ is the mass of point $P_{ijk}$, $\vec{a}_{ijk}$ is its acceleration, and $\vec{F}_{ijk}$ is the net force on $P_{ijk}$.

The elastic force is described by Hooke's law:

$$\vec{F}_{elastic}(P_{ijk}) = \sum_{P_{uvw} \in Set_8} k\left(\left|P_{ijk}P_{uvw}\right|_t - \left|P_{ijk}P_{uvw}\right|_0\right) \frac{\overrightarrow{P_{ijk}P_{uvw}}}{\left|P_{ijk}P_{uvw}\right|_t} \quad (2)$$

where $k$ is the elastic coefficient of the springs, $\left|P_{ijk}P_{uvw}\right|_t$ is the distance from $P_{ijk}$ to $P_{uvw}$ at time $t$, $\left|P_{ijk}P_{uvw}\right|_0$ is the initial distance, $\text{Set}_8$ denotes the set of 8 adjacent points of $P_{ijk}$.

The damping force is given by

$$\begin{aligned} \vec{F}_{damp} &= -C_{damp} \frac{\overrightarrow{\Delta P_{ijk}(t)}}{\Delta t} \\ &\approx -C_{damp} \vec{V}_{ijk}(t) \end{aligned} \quad (3)$$

where $C_{damp}$ is the damping coefficient and $\frac{\overrightarrow{\Delta P_{ijk}(t)}}{\Delta t}$ is an approximation of the velocity $\vec{V}_{ijk}(t)$ of mass point $P_{ijk}$ at time $t$.

The virtual force can be described by

$$\begin{aligned} \vec{F}_{virtual}(P_{ijk}) &= -K'' \times \overrightarrow{\Delta S} \\ &= -K'' \times \{x_i(t) - x_i(0), y_j(t) - y_j(0), z_k(t) - z_k(0)\} \end{aligned} \quad (4)$$

where $K''$ is the elastic coefficient of the virtual springs and $\overrightarrow{\Delta S}$ is the displacement of the control point from its initial position.

The surgical force is given by

$$\vec{F}_{surgical}(P_{ijk}) = \vec{F}_{exerting} \times e^{-dist(P_{ijk}) \times 150 - 10} \quad (5)$$

### D. Verlet Integration

To calculate the new position of a mass point $P_{ijk}$, we apply Taylor's expansion:

$$P_{ijk}(t + \Delta t) = P_{ijk}(t) + \Delta t \cdot \vec{v}_{ijk}(t) + (1/2) \cdot \Delta t^2 \cdot \vec{a}_{ijk}(t) + o(\Delta t^3) \quad (6)$$

$$P_{ijk}(t - \Delta t) = P_{ijk}(t) - \Delta t \cdot \vec{v}_{ijk}(t) + (1/2) \cdot \Delta t^2 \cdot \vec{a}_{ijk}(t) + o(\Delta t^3) \quad (7)$$

$$P_{ijk}(t + \Delta t) \approx 2P_{ijk}(t) - P_{ijk}(t - \Delta t) + \Delta t^2 \cdot \vec{a}_{ijk}(t) \quad (8)$$

Let us denote the acceleration $\vec{a}_{ijk}$ at time $t$ by $(a_i(t), a_j(t), a_k(t))$, then we can calculate the position of point $P_{ijk}$ $(x_i, y_j, z_k)$ at the next time point:

$$\left\{ \begin{array}{l} x_i(t + \Delta t) \approx 2x_i(t) - x_i(t - \Delta t) + \Delta t^2 a_i(t) \\ y_j(t + \Delta t) \approx 2y_j(t) - y_j(t - \Delta t) + \Delta t^2 a_j(t) \\ z_k(t + \Delta t) \approx 2z_k(t) - z_k(t - \Delta t) + \Delta t^2 a_k(t) \end{array} \right\} \quad (9)$$

where Δt is a chosen time interval.

### E. Image Rending

We employed the bilinear interpolation to process the grid of mass points. Bilinear interpolation is an extension of linear interpolation for interpolating functions of two variables on a regular grid. The key idea is to perform linear interpolation first in one direction, and then in the other direction.

The main idea of the interpolation of mesh of mass points is that smaller mesh matrices of mass points in the 3D mesh deformation procedure are interpolated into larger mesh matrices. For example, we can interpolate 31×31×3 mesh matrices into larger mesh matrices in 3D space (see Figure 2, Figure 3 and Figure 4).

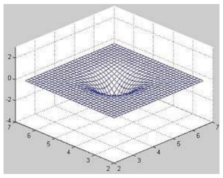

Figure 2.

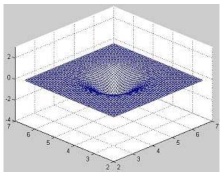

Figure 3.

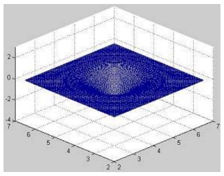

Figure 4.

Figure 2 is 31 ×31 meshes before interpolation. Figure 3 is 61×61 meshes after bilinear interpolation when one mass point is added between two adjacent mass points in Figure 2. Figure 4 is 121×121 meshes after bilinear interpolation when three mass points are added between two adjacent mass points in Figure 2.

From above example, we find that the interpolation of deformation mesh matrices of mass points may be divided into 2 steps.

- n mass points are added between two adjacent mass points in the deformation mesh matrices of m×m×3 size. After adding n new mass points, the size of new mesh matrices is M×M×3 (M=(m-1)*n+m).
- According to m×m×3 mesh matrices, we can employ bilinear interpolation to interpolate respectively the X, Y and Z matrices of coordinates of new added mass points in 3D space.

Assume that 10 mass points are added between two adjacent mass points in the deformation mesh matrices of 31×31 size. We first obtain 331×331 mesh matrix by bilinear interpolation to interpolate 31×31 deformation mesh matrix. And then the colors of 331×331 pixels in the endoscopic image are mapped and filled in 331×331 mesh matrix to finish the image rendering. As shown in Figure 5, we use one endoscopic image to do the image rendering according to the above method.

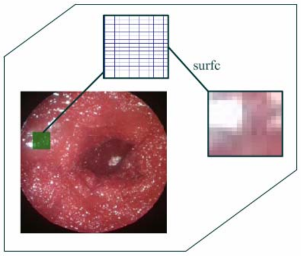

Figure 5. Endoscopic image rendering

## III. EXPERIMENTS

We first simulated the exerting forces. Normally, $\vec{F}_{surgical}(P_{ijk})$ decreases quickly as the distance between $P_{ijk}$ and the control point increases.

We adopted an exponential expression:

$$\sum \vec{F}_{surgical}(P_{ijk}) \approx \sum \vec{F}_{exerting} \times e^{-dist(P_{ijk})\times c+c'}$$

We experimentally chose c′=-10 and c=150. The parameters in Equations (2)~(8) were also chosen experimentally. Given a fixed surgical force, we simulated the deformation process of the control point. The displacement of the control point from its initial position is plotted by varying the value of *k*. It can be observed that noticeable oscillation may occur at large *k*, which is undesirable. In our experiment, we found that *k*=10 was appropriate. The damping force has a buffering effect which prevents abrupt deformation. Following the same process, we chose *k*=10, $C_{damp}$=0.5, $K''$=5.

We implemented the model with Visual C++ and OpenGL and simulated tissue deformation on a personal computer. By exerting a force $\vec{F}_{exerting} = (F_{dx}, F_{dy}, F_{dz})$ on a flat tissue surface, we computed the response using our 3-D spring-mass model. The deformation was rendered in real-time. The results were highly satisfactory in direct visualization. Some typical results displayed in wire meshes are shown in Figure 6, Figure 7 and Figure 8.

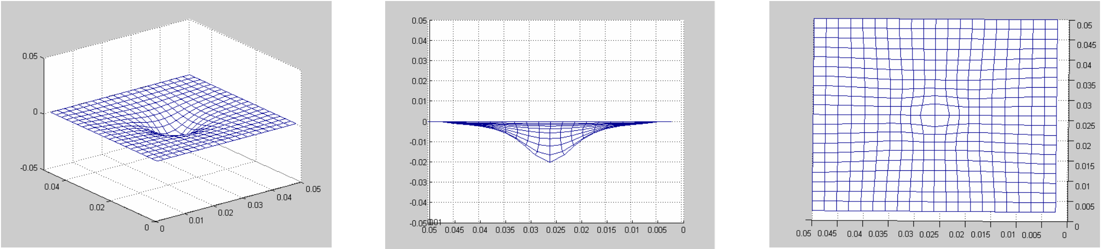

Figure6. Mesh dynamic deformation of vertical force (60% vertical force)

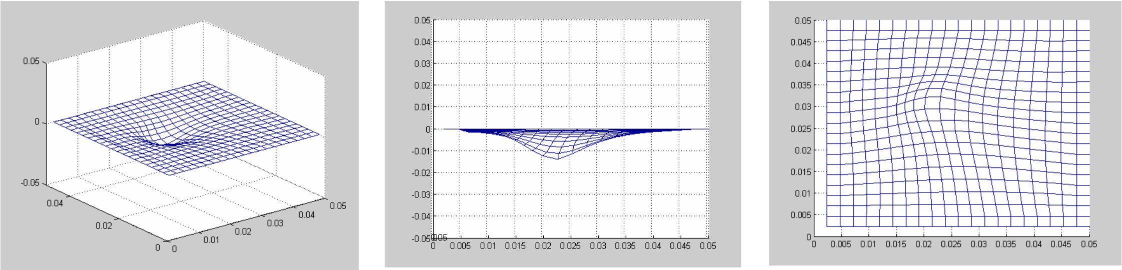

Figure7. Mesh dynamic deformation of vertical force (60% oblique force)

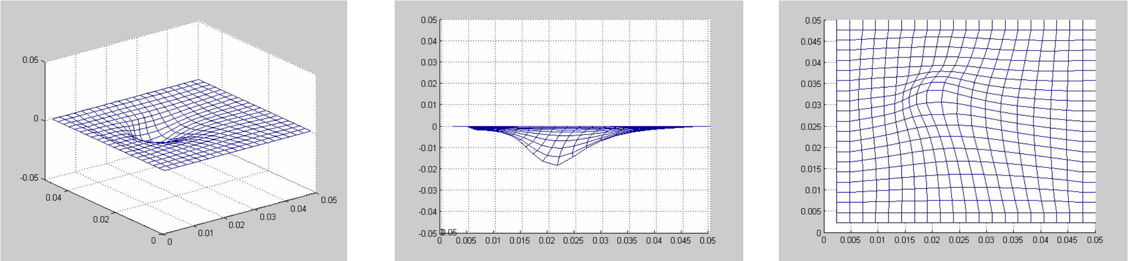

Figure8. Mesh dynamic deformation of vertical force (100% oblique force)

## IV. CONCLUSIONS

We have presented an improved mass-spring model to simulate soft tissue deformation for simulation. In addition to the conventional springs, we have proposed to use a virtual spring to model the resistant force along the direction of the exerting force. We applied the Verlet integration to calculate the position of a mass point without explicitly computing related velocity values. Adding mass points between two adjacent mass points and interpolating the new added mass points by bilinear interpolation were used for image rendering. Model parameters are determined via simulation experiments and may be further improved.

As a result, the deformed mesh can be updated in real-time. Our study provides an important component to construct the real-time and dynamic simulators.

## ACKNOWLEDGMENT

The research work described in this paper was fully supported by one grant from the National Natural Science Foundation of China (Project No. 60703070). The authors would like to thank Dr. Mingui Sun, Dr. Qiang Liu and Dr. Sclabassi at the Laboratory of Computing Neuroscience in University of Pittsburgh for directing the medical image processing and surgical simulation research work.

## REFERENCES

[1] Cagetay Basdogn et al., “VR-Based Simulators for Training in Minimally Invasive Surgery”, IEEE Computer Graphics and Applications, 2007, Vol. 27, Issue 2, pp.54-66.

[2] Andre Neubauer, Stefan Wolfsberger, Marie-Therese Forster, Lukas Mroz et al., “Advanced Virtual Endoscopic Pituitary Surgery”, IEEE Transactions on Visualization and Computer Graphics, 2005, vol.11, no. 5, pp.497-507.

[3] U.Kuhnapfei, H.K. Cakmak, and H.Maab, “Endoscopic Surgery training Using Virtual reality and Deformable tissue Simulation”, Computer & graphics, 2000, vol.24, no.5, pp.671-682.

[4] J Weil, “The synthesis of cloth objects”, Computer graphics, 1986, 20(4), pp.49-54.

[5] N J Gu, H C Shen, et al., “Computer Modeling, analysis, and synthesis of dressed humans”, IEEE Transactions on Circuits and Systems for Video Technology, 1999, 9(2), pp.378-388.

[6] J Lander., “Devil in the blue-faceted dress: Real-time cloth animation”, Game Developer, 1999, 6(5), pp.17-21.

[7] S Gray, “In virtual fashion”, IEEE Spectrum, 1998, 35(2), pp.18-25.

[8] P Volino, N M Thalmann, J Shen, et al., “An evolving system for simulating clothes on virtual actors”, IEEE Computer Graphics & Applications, 1996, 16(4), pp.42-50.

[9] Verlet Loup, “Computer Experiments on Classical Fluids”, Physical Review, 1967, 159(1), pp.98-103.

[10] X Provot, “Deformation constraints in a mass-spring model to describe rigid cloth behavior”, Proceedings of Graphics Interface, Quebec: Canadian Information Processing Society, 1995, pp147-154.