A Microsurgery Simulation System

Joel Brown[1], Kevin Montgomery[2], Jean-Claude Latombe[1], and Michael Stephanides[2]

[1] Computer Science Department, Stanford University
[2] Department of Surgery, Stanford University

Abstract. Computer systems for surgical planning and training are poised to greatly impact the traditional versions of these tasks. These systems provide an opportunity to learn surgical techniques with lower costs and lower risks. We have developed a virtual environment for the graphical visualization of complex surgical objects and real-time interaction with these objects using real surgical tools. An application for microsurgical training, in which the user sutures together virtual blood vessels, has been developed. This application demonstrates many facets of our system, including deformable object simulation, tool interactions, collision detection, and suture simulation. Here we present a broad outline of the system, which can be generalized for any anastomosis or other procedures, and a detailed look at the components of the microsurgery simulation.

1 Introduction

As computer power and graphics capabilities continue to increase, there is growing interest in surgical simulation as a technique to enhance surgeons' training. Such training currently requires cadavers or laboratory animals. A computer simulation option could reduce costs and allay ethical concerns, while possibly decreasing training time and providing better feedback to the trainees. However, for surgical simulation to be useful it must be realistic with respect to tissue deformation, tool interactions, visual rendering, and real-time response. This paper describes a microsurgery training system based on novel computer simulation techniques. The system allows a user to interact with models of deformable tissues using real surgical instruments mounted on trackers. It generates a graphic rendering of the tissue deformations in real-time. Its key components are new algorithms for fast and realistic simulation of tissue and suture, and for detecting and processing contacts among rigid and deformable objects.

1.1 Related Work

Research on modeling soft-tissue deformation has increased dramatically in the past few years, with focus on physically-based models for simulation. Terzopoulos and Waters [1] argue the advantages of using anatomy and physics rather than just geometry for facial animation, and present a mass-spring model of

W. Niessen and M. Viergever (Eds.): MICCAI 2001, LNCS 2208, pp. 137–144, 2001.

facial tissue with muscle actuators. Joukhadar and Laugier [2] also use a mass-spring model with explicit integration techniques as the foundation of a general dynamic simulation system, and Baraff and Witkin [3] use masses and springs with implicit integration to simulate cloth. These mass-spring models are characterized by fast computation and simple implementation [10]. They can model in great geometric detail tissues that have nonlinear, non-homogeneous, and anisotropic visco-elastic properties.

Finite element models (FEMs) have been used in order to more rigorously capture biomechanical properties of human tissues. Works in [4,5] use FEMs to model facial tissue and predict surgery outcomes. The increased computational demands of finite elements are serious hurdles for real-time simulation. Numerical techniques, including pre-computation of key deformations, are proposed in [6,7] to significantly reduce computation. The endoscopic training tool of [8] describes a system which uses either mass-spring or FEM depending on the situation. The hybrid elastic approach of [11] combines aspects of both models in a simulation which allows deformation and cutting of tissues in real-time.

There are many other examples of mass-spring models and FEMs, as well as some alternate models, too numerous to cite them all here. The consensus is that FEMs can be very biomechanically accurate, but are not always appropriate for large deformations, or for real-time simulation of large geometries. Conversely, determining the proper parameters for mass-spring models can be very difficult, as can be determining proper placement of masses and springs to adequately model an object's volume. Surgical training simulation tends to rely more on visual realism than exact, patient-specific deformation (which may be more necessary in planning and predicting a specific patient's surgery, for example), and thus we have focused our research on mass-spring models, with an eye towards computation reduction for displaying complex virtual environments.

The need for suture simulation has been previously addressed in [12], and a performance study in [13] discusses the validity of using such a simulator to develop surgical skill, although it does not provide technical details of the actual simulation. More recently, [8] discusses many different aspects of suture simulation.

Interaction with virtual tools is another necessary component of any realistic simulator, and has attracted a fair amount of recent attention [9]. Surgical tools must not only be modeled accurately, but also pull, push, grasp, and cut other objects, often causing deformations by their actions. The first step of simulating such interactions is accurate collision detection, which has been studied extensively for rigid objects [16], although not nearly as much for deformable models. After detection, novel algorithms to handle collision response need to be developed to allow for a wide range of surgical actions [2].

1.2 Description of Microsurgery

Microsurgery is a well-established surgical field which involves the repair of approximately 1mm vessels and nerves under an operating microscope. It is a

necessity in many reconstructive procedures, including the successful reattachment of severed digits. Using a forceps, the surgeon maneuvers a suture through the ends of two vessels, loops the suture around a second forceps, and pulls it tightly through itself to knot the vessels together. With several such stitches, the severed vessel can be repaired. Microsurgeons typically acquire their initial skills through months of practice in an animal lab, at which point they still require months of supervision in the operating room. Without practice, these skills can quickly degrade.

2 System Overview

Our software system includes a deformable object simulator, a tool simulator, and a collision detection module. A graphics display allows any number of objects to be rendered from a 3D virtual world onto the screen at 30 Hz or more. The user has complete control of the view, and may use stereo glasses for true binocular depth perception. The positions of the objects are read from the deformable object and tool simulators before each screen refresh.

Deformable object simulation is described in detail in the following two sections. Tool simulation synchronizes virtual surgical tools with real tools that are connected to external tracking devices. Virtual tools consist of one or more rigid parts modeled as triangulated surfaces. Their positions and orientations are controlled by the external devices at high update rates (typically 100 Hz), and other information from the devices may control relative rotations or translations of the parts that make up one tool. Interactions between tools and deformable objects, such as grabbing, poking, and cutting, are dependent on the collision detection module (Section 5).

The system also supports parallel processing using multithreading. Two separate threads of execution allow the simulation and collision detection to not conflict with the display. In this way, visual updates occur at a guaranteed rate while the simulation continues uninterrupted.

The setup for microsurgery includes two real surgical forceps instrumented to detect closure and attached to electromagnetic trackers (miniBIRD, Ascension Technology Corporation). The user's translation, rotation, opening, and closing of these forceps directly controls forceps models in the simulation. Using the forceps, models of blood vessels can be grabbed and deformed. A suture (needle and thread) can be manipulated to pierce through and realistically interact with the vessels and the forceps. Stereo glasses allow the necessary depth perception to complete the task. Figure 1*a* shows a user of the simulator.

3 Soft-Tissue Modeling

We represent the volumetric geometry of a deformable object by a 3D mesh M of n nodes N_i ($i = 1,...,n$) connected by links L_{ij}, $i, j \in [1, n], i \neq j$. The nodes and links are grouped into triangles on the surface, for graphics purposes, but

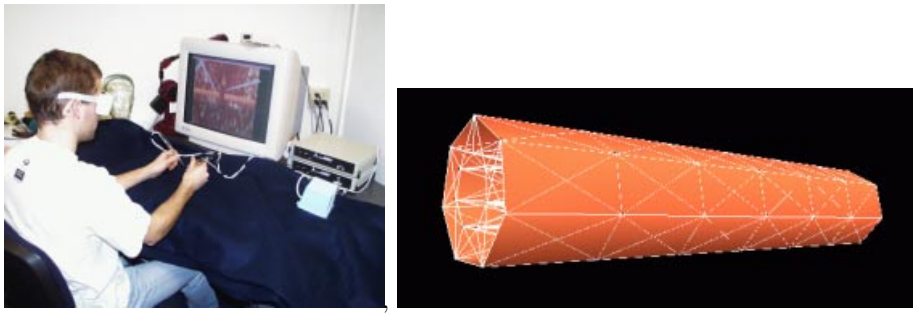

Fig. 1. (a) Setup for microsurgery (b) Shaded vessel shown with links

unrestricted below the surface. Each node maps to a specific point of the object, so that the displacements of the nodes describe the deformation of the object.

The viscoelastic properties of the object are described by additional data stored in the nodes and links of M. More precisely, a mass m_i and a damping coefficient c_i are associated with each node N_i, and a stiffness k_{ij} is associated with each deformable link L_{ij}. The internal force between N_i and N_j is $\boldsymbol{F}_{ij} = -k_{ij}\Delta_{ij}\boldsymbol{u}_{ij}$, where $\Delta_{ij} = l_{ij} - rl_{ij}$ is the current length of the link minus its resting length, and $\boldsymbol{u}_{ij}$ is the unit vector pointing from N_i toward N_j. The stiffness k_{ij} may be constant or a function of Δ_{ij}, and in both cases, $\boldsymbol{F}_{ij}$ is a function of only the coordinate vectors $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ of N_i and N_j. This representation makes it possible to describe objects that have nonlinear, non-homogeneous, and anisotropic properties.

At any instant of time t, the motion and deformation of M are described by a system of n second-order differential equations, each expressing the motion of a node N_i:

$$m_i\boldsymbol{a}_i + c_i\boldsymbol{v}_i + \sum_{j\in\sigma(i)} \boldsymbol{F}_{ij}(\boldsymbol{x}_i, \boldsymbol{x}_j) = m_i\boldsymbol{g} \tag{1}$$

where $\boldsymbol{x}_i$ is the coordinate vector of N_i, $\boldsymbol{v}_i$ and $\boldsymbol{a}_i$ are its velocity and acceleration vectors, respectively, and $m_i\boldsymbol{g}$ is the gravitational force. $\sigma(i)$ denotes the set of the indices of the nodes in M connected by links to N_i. Some nodes may be fixed in space, or directly manipulated by surgical tools, in which case their positions are read from memory or computed from the positions of the tracking devices, rather than via (1).

To dynamically simulate the modeled tissue, we have implemented several numerical integration techniques to solve (1), including forward Euler, and second and fourth order Runge-Kutta solvers. These are all based on advancing the simulation by an amount of time Δt, and using the position, velocity, and force information for each node at time t to find positions and velocities at time $t + \Delta t$.

We have also developed a faster "quasi-static" algorithm, based upon the assumptions that the velocity of user-displaced nodes is small enough and damping large enough that the mesh achieves static equilibrium at each instant. These as-

sumptions are reasonable for many human-body tissues and surgical operations on these tissues. Under these assumptions we can neglect dynamic inertial and damping forces, and compute the current shape of M by solving this system of equations:

$$\sum_{j \in \sigma(i)} \boldsymbol{F}_{ij}(\boldsymbol{x}_i, \boldsymbol{x}_j) - m_i \boldsymbol{g} = \boldsymbol{0} \tag{2}$$

The reduced computation can be a significant savings in the many situations where the quasistatic assumptions are appropriate.

We let I be the set of indices of all variable nodes (those which are not fixed in space or directly grasped by the user), and let $\boldsymbol{f}_i$ be the total force acting on each variable node, $\boldsymbol{f}_i = \sum_{j \in \sigma(i)} \boldsymbol{F}_{ij} - m_i \boldsymbol{g}$. An iterative algorithm for solving (2) at real-time animation rates is as follows:

1. Compute positions of user-displaced nodes from positions of tracking devices
2. Repeat until 1/30 sec. has elapsed:
   For every $i \in I$:
   (a) Update $\boldsymbol{f}_i$ based on (2): $\boldsymbol{f}_i \leftarrow \sum_{j \in \sigma(i)} \boldsymbol{F}_{ij} - m_i \boldsymbol{g}$
   (b) Update $\boldsymbol{x}_i$ based on the force $\boldsymbol{f}_i$: $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i + \alpha \boldsymbol{f}_i$

Ideally, the value of α is chosen as large as possible such that the iteration converges, and the values $\boldsymbol{f}_i$ approach $\boldsymbol{0}$ before each redraw. This choice is typically determined by experimental trials. By performing rendering in a separate thread, the entire time interval is used for computing the equilibrium positions $\boldsymbol{x}_i$ of the nodes. By ordering the indices in I in a breadth-first manner starting at the user-displaced nodes, and proceeding along links in the mesh, forces converge faster. In addition, we can handle objects with many more nodes by limiting the deformation region to only those nodes within a certain distance of the user-displaced nodes, or by cutting off propagation when forces drop below a certain threshold.

The vessels in our simulation are modeled as double-hulled cylinders, with the inner and outer cylinders representing the thickness of the vessel. Each cylinder consists of several layers of nodes, with the layers evenly spaced, and each layer consists of several nodes evenly spaced around a circle. Each node is connected by deformable links to its neighbors within a layer and in neighboring layers. There are also connections between the inner and outer cylinders, which provide torsional stability, preventing the vessel from twisting around its long axis. The end layers of each vessel are fixed in space, representing the fact that the vessels are clamped down during surgery, and only a portion of their length can be manipulated. Figure 1*b* shows a smooth-shaded vessel and its underlying links.

As the user displaces individual nodes of the vessels, the quasi-static algorithm described above is used to calculate the deformation. Figures 2*a* and 2*b* show some examples of deforming the vessels with forceps.

4 Simulation of the Suture

The suture is deformable but not elastic, so the above deformation techniques based on mass-spring models are not applicable. Instead it should behave as a

needle and thread, which can stretch minimally if at all, and has a free-form shape which is affected by gravity and direct contacts. To achieve realistic deformation, we model the suture as an articulated object: many short, linear links (edges) are connected together at nodes which act as spherical joints. The joints allow two degrees of rotational freedom, while the edges are rigid and short enough that the suture shape appears smooth. By keeping the angles between the first few edges fixed, we can model a rigid needle at one end of the suture.

To model the motion of the suture, constraint-based techniques are used. Any node of the suture may be constrained by another object in the system. For example, one node might be grasped by a forceps, and thus its position is constrained by the forceps. If the suture has pierced through a vessel, a node will be constrained by the position of the vessel. Finally, if the suture is draped over another object, nodes will be constrained by that object.

The motion is then calculated in a "follow-the-leader" manner as follows: a constrained node N_i is moved by the constraining object from $\boldsymbol{x}_{i.old}$ to $\boldsymbol{x}_{i.new}$. Its neighbor N_{i+1} then computes its new position $\boldsymbol{x}_{i+1.new}$ as the point a distance d along the line from $\boldsymbol{x}_{i.new}$ to $\boldsymbol{x}_{i+1.old}$, where d is the (fixed) length of the edge connecting N_i and N_{i+1}. The same is done for node N_{i-1}. This motion is propagated up and down the suture to N_{i+1}, N_{i+2}, ... and N_{i-1}, N_{i-2}, ... until the next constrained node or one end of the suture is reached. For nodes between two constrained nodes N_i and N_j, the preceding algorithm will compute two preliminary results, propagating from N_i to N_j, and from N_j to N_i. These results are averaged to give the final position.

Certain constraints are designated as *soft*, such as where the suture is piercing a vessel or draped over another object, whereas the forceps grabbing the suture is a *hard* constraint. The distinction is that the suture can slide over and/or through a soft constraint, changing which node of the suture is constrained. It may be the case that two constraints move in opposing directions and cause the suture to stretch between them. In that case, the suture will slide through a soft constraint to decrease the stretch, but will break if between two hard constraints (in real surgery, it is not difficult to break the suture by pulling with both forceps in opposite directions). Additionally, if the suture is pierced through a vessel and is pulled on both ends, the suture will pull the vessel, causing it to deform as in Fig. 2*c*. Figure 2*d* shows the suture pulling together the two vessels.

5 Collisions and Interactions

Almost all object interactions depend at some level on collision detection [16]. Grabbing is achieved by finding nodes colliding with the tip of the grabbing object (e.g. forceps). Piercing the vessel requires finding a collision between the needle edges and a vessel face. Draping the suture around another object also involves edge to face collisions (Figures 2*e* and 2*f* show the suture around forceps and vessel), and draping it around itself requires edge to edge self-collisions. Other interactions modeled by the system (although not specifically in the mi-

crosurgery simulation) include prodding one object with another (face to face collisions), and cutting one object with another (edge to face collisions).

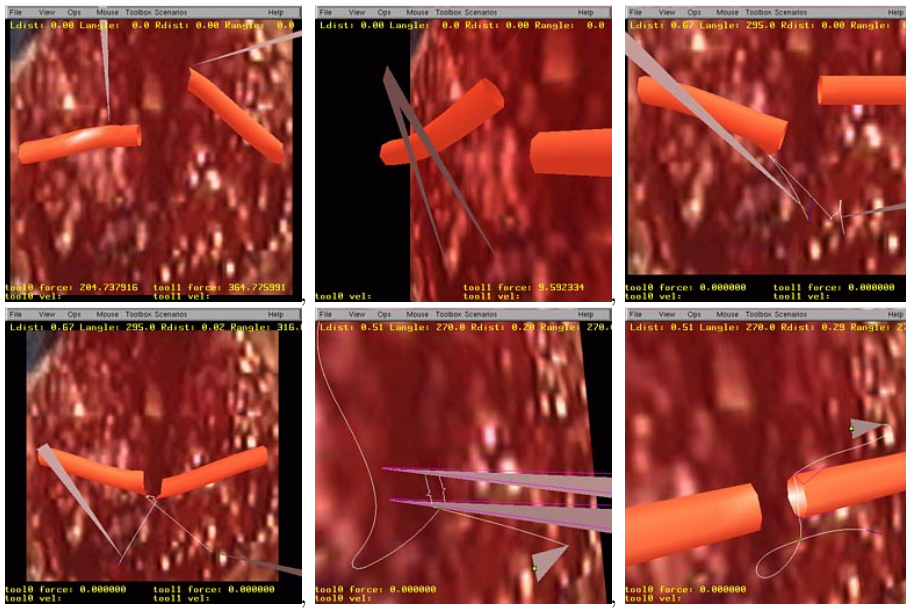

Fig. 2. (a)-(f): Scenarios for vessel, forceps, and suture interaction

The scheme we use for collision detection and distance computation is based on Quinlan's bounding sphere hierarchy [14]. This algorithm was extended by Sorkin [15] to allow for collisions between deforming objects by efficiently updating the bounding sphere hierarchies as these objects deform. We can comfortably find all collisions between forceps, vessels, and suture (including internal edge to edge collisions in the suture) at every redraw, without affecting the animation rate of the simulation.

6 Conclusions

By combining tool interactions, tissue deformation, collision detection, and high-resolution graphics, we have created a preliminary microsurgery simulation which has been exhibited to many plastic and reconstructive surgeons, and deemed realistic and potentially very useful. Our next step is experimental and clinical verification, by having surgeons who are learning the procedure use this tool, and assessing the quality of their virtual repairs through measurements such as angle and position of vessel piercing. We will then try to establish quantitatively how practicing with the simulator affects future quality of real vessel repairs.

Acknowledgements: This work was supported by NASA (NAS-NCC2-1010), NSF (IIS-99-07060-001), and NIH National Libraries of Medicine (NLM-3506). Special thanks also to Cynthia Bruyns, Frederic Mazzella and Stephen Sorkin for their contributions to this project.

References

1. D. Terzopoulos and K. Waters. Physically-Based Facial Modelling, Analysis, and Animation. *J. of Visualization and Computer Animation* Vol 1: 73-80, 1990.
2. A. Joukhadar and C. Laugier. Dynamic Simulation: Model, Basic Algorithms, and Optimization. In *Algorithms For Robotic Motion and Manipulation*, J. Laumond and M. Overmars (eds.), A.K. Peters Publisher, pp. 419-434, 1997.
3. D. Baraff and A. Witkin. Large Steps in Cloth Simulation. *ACM SIGGRAPH 98 Conference Proceedings*, pp. 43-52, 1998.
4. R. Koch, M. Gross, F. Carls, D. von Büren, G. Fankhauser, and Y. Parish, Simulating Facial Surgery Using Finite Element Models. *ACM SIGGRAPH 96 Conference Proceedings*, pp. 421-428, 1996.
5. S. Pieper, D. Laub, and J. Rosen. A Finite-Element Facial Model for Simulating Plastic Surgery. *Plastic and Reconstructive Surgery*, 96(5): 1100-1105, Oct 1995.
6. M. Bro-Nielsen and S. Cotin. Real-time Volumetric Deformable Models for Surgery Simulation using Finite Elements and Condensation. *Computer Graphics Forum*, 15(3): 57-66 (Eurographics '96), 1996.
7. J. Berkley, S. Weghorst, H. Gladstone, G. Raugi, D. Berg, and M. Ganter. Fast Finite Element Modeling for Surgical Simulation. *Proceedings of Medicine Meets Virtual Reality 1999*, pp. 55-61, 1999.
8. U. Kühnapfel, H. K. Çakmak, H. Maaß. Endoscopic Surgery Training Using Virtual Reality and Deformable Tissue Simulation. *Computers & Graphics*, Volume 24: 671-682, 2000.
9. C. Basdogan. Simulation of Instrument-Tissue Interactions and System Integration. *Medicine Meets Virtual Reality (MMVR2001)*, Newport Beach, CA, January 27, 2001, *http://eis.jpl.nasa.gov/˜basdogan/Tutorials/MMVRTuto01.pdf*
10. S. Cotin, H. Delingette, and N. Ayache. Real-time Elastic Deformations of Soft Tissues for Surgery Simulation. *IEEE Transactions On Visualization and Computer Graphics*, 5(1): 62-73, January-March 1999.
11. S. Cotin, H. Delingette, and N. Ayache. A Hybrid Elastic Model Allowing Real-Time Cutting, Deformations and Force-Feedback for Surgery Training and Simulation. *The Visual Computer*, 16(8): 437-452, 2000.
12. H. Delingette. Towards Realistic Soft Tissue Modeling in Medical Simulation. *Proc. of the IEEE : Special Issue on Surgery Simulation*, pp. 512-523, April 1998.
13. R. O'Toole, R. Playter, T. Krummel, W. Blank, N. Cornelius, W. Roberts, W. Bell, and M. Raibert. Measuring and Developing Suturing Technique with a Virtual Reality Surgical Simulator. *J. of the American College of Surgeons*, 189(1): 114-127, July 1999.
14. S. Quinlan. Efficient Distance Computation Between Non-Convex Objects. *Proc. IEEE Int. Conf. On Robotics and Automation*, pp. 3324-3329, 1994.
15. S. Sorkin. *Distance Computing Between Deformable Objects.* Honors Thesis, Computer Sc. Dept., Stanford University, June 2000.
16. M. Lin and S. Gottschalk. Collision Detection Between Geometric Models: A Survey. *Proc. of IMA Conference on Mathematics of Surfaces*, pp. 37-56, 1998.