12-2008

Real-Time Graphic and Haptic Simulation of Deformable Tissue Puncture

Joseph Romano
University of Pennsylvania, jrom@seas.upenn.edu

Alla Safonova
University of Pennsylvania, alla@cis.upenn.edu

Katherine J. Kuchenbecker
University of Pennsylvania, kuchenbe@seas.upenn.edu

Real-Time Graphic and Haptic Simulation of Deformable Tissue Puncture

Abstract
A myriad of surgical tasks rely on puncturing tissue membranes (Fig. 1) and cutting through tissue mass. Properly training a practitioner for such tasks requires a simulator that can display both the graphical changes and the haptic forces of these deformations, punctures, and cutting actions. This paper documents our work to create a simulator that can model these effects in real time. Generating graphic and haptic output necessitates the use of a predictive model to track the tissue's physical state. Many finite element methods (FEM) exist for computing tissue deformation ([1],[4]). These methods often obtain accurate results, but they can be computationally intensive for complex models. Real-time tasks using this approach are often limited in their complexity and workspace domain due to the large computational overhead of FEM. The computer graphics community has developed a large range of methods for modeling deformable media [5], often trading complete physical accuracy for computational speedup. Casson and Laugier [3] outline a mass-spring mesh model based on these principles, but they do not explore its usage with haptic interaction. Gerovich et al. [2] detail a set of haptic interaction rules (Fig. 2) for one dimensional simulation of multi-layer deformable tissue, but they do not provide strategies for integrating this model with realistic graphic feedback.

Disciplines
Computer Sciences | Engineering | Graphics and Human Computer Interfaces

Real-Time Graphic and Haptic Simulation of Deformable Tissue Puncture

Joseph M. Romano, Alla Safonova, and Katherine J. Kuchenbecker
University of Pennsylvania, GRASP Laboratory and HMS Center
{jrom, alla, kuchenbe}@seas.upenn.edu

Background

A myriad of surgical tasks rely on puncturing tissue membranes (Fig. 1) and cutting through tissue mass. Properly training a practitioner for such tasks requires a simulator that can display both the graphical changes and the haptic forces of these deformations, punctures, and cutting actions. This paper documents our work to create a simulator that can model these effects in real time.

Generating graphic and haptic output necessitates the use of a predictive model to track the tissue's physical state. Many finite element methods (FEM) exist for computing tissue deformation ([1],[4]). These methods often obtain accurate results, but they can be computationally intensive for complex models. Real-time tasks using this approach are often limited in their complexity and workspace domain due to the large computational overhead of FEM.

The computer graphics community has developed a large range of methods for modeling deformable media [5], often trading complete physical accuracy for computational speedup. Casson and Laugier [3] outline a mass-spring mesh model based on these principles, but they do not explore its usage with haptic interaction. Gerovich et al. [2] detail a set of haptic interaction rules (Fig. 2) for one-dimensional simulation of multi-layer deformable tissue, but they do not provide strategies for integrating this model with realistic graphic feedback.

Tools and Methods

Our simulation uses mass-spring meshes similar to those developed in [3] and implements the force rules for contact and puncture from [2]. We create three planar meshes, one for each tissue layer, which can then be deformed and punctured. The graphical output of our model is displayed to the user via an OpenGL window (Fig. 3). The user interacts with the mesh models via a Phantom Omni haptic device (Fig. 4); its motion drives the movement of the virtual needle, and its motors display contact forces to the user. The current implementation keeps the needle vertical at all times, but future versions will allow contact at any angle.

Our software program performs its three main tasks (mesh update, graphic output, and haptic output) at different rates to achieve optimal performance for the user (Fig. 5). One thread updates the motion of the mesh models as quickly as possible, using a fourth-order Runge-Kutta integration algorithm. A simulation with 1875 total nodes was able to run at approximately 55 Hz on a modern desktop computer. This computational rate easily enables smooth graphical output at 30 Hz.

In order to prevent discrete changes in force from being noticed by the user, haptic devices must update their force output at a rate exceeding human sensing capabilities (~600 Hz). Since our mesh model updates at only ~55 Hz, the 1 kHz haptic thread treats the mesh node positions as static between model updates. Using this approach we can provide smooth haptic feedback even though calculating the actual mesh deformation at such rates is not possible. The program computes the haptic output force by summing the simple position- and velocity-based terms from Fig. 2 with the net normal force acting on the closest mesh node, smoothed via a low-pass filter. Between mesh updates, the forces generated from the Fig. 2 rules are summed and then applied together to the appropriate nodes during the mesh's next RK4 integration step. A layer is punctured if the contact node experiences a net normal force beyond a critical value.

Results

The application of damping forces to the mesh model results in graphically pleasing effects at the tissue surfaces. The surface appears to 'stick' to the needle as it is inserted and retracted from the model, and a haptically realistic viscous damping sensation is created.

Datasets of the user's position and applied force were recorded for several interactions with our simulation (Fig. 6). Distinct 'popping' actions can be felt when tissue layers are broken, as expected when using the rules seen in Fig. 2.

Our simulation was executed on several different computing platforms in order to test the combined effect of the haptic and graphic interaction on commonly available computing resources, as seen in Fig. 7.

Conclusion

Our new simulation techniques are suitable for immediate implementation on available computing hardware. The reaction of the model is similar to physical needle puncture data taken by other groups such as [2]. We believe that our method of rendering haptic and graphical output from a mass-spring model can enable today's simulations to perform the real-time dynamic reactions that they are lacking. Although we have not yet performed a formal user study, we believe this enhanced output will be of considerable use to the simulation community.

References

[1] S. P. DiMaio and S. E. Salcudean. Needle insertion modeling and simulation. *IEEE Transactions on Robotics and Automation* 19(5):864–875, October 2003.

[2] O. Gerovich, P. Marayong, and A. M. Okamura. The effect of visual and haptic feedback on computer-assisted needle insertion. *Computer-Aided Surgery*, 9(6):243–249, 2004.

[3] F. Boux de Casson and C. Laugier. Modeling the dynamics of a human liver for a minimally invasive surgery simulator. *MICCAI*, pages 1156–1165, 1999.

[4] M. Mahvash and V. Hayward. Haptic rendering of cutting: A fracture mechanics approach. *Haptics-e*, www.haptics–e.org, November 2001.

[5] D. Baraff and A. Witkin. Large steps in cloth simulation. *25th Annual Conference on Computer Graphics and Interactive Techniques SIGGRAPH*, pages 43–54, 1998.

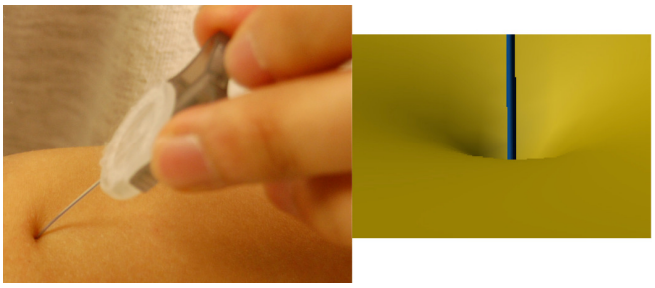

Fig. 1: Real skin deformation (left) and our simulator deformation (right).

Tissue Layer	State	Force
Skin	Pre-Puncture	$F = k_s y$
	Post-Puncture	$F = b_s y v$
Fat	Pre-Puncture	$F = k_f y + b_s y_s v$
	Post-Puncture	$F = (b_f y + b_s y_s)\, v$
Bone	Pre-Puncture	$F = k_b y + (b_f y_f + b_s y_s)\, v$

Fig. 2: Tissue force response rules adapted from [2]. Pre-Puncture is defined as the regime where the needle has contacted the tissue surface but not yet broken through. Post-Puncture is the regime after breaking the previous tissue surface, but prior to contacting the next. y represents the depth of the haptic device tip with respect to the current surface; y_f, y_s the current thickness of each tissue layer, and v the velocity of the haptic device tip.

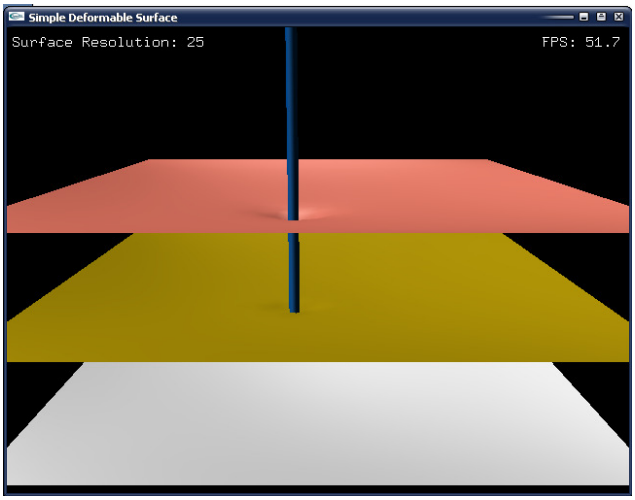

Fig. 3: Sample graphical output. The user-controlled needle is shown in blue. A damping force is being applied to the first 'skin' layer that has already been punctured, while the user begins to deform the second 'fat' layer. The user feels the reaction force as the tissue deflects.

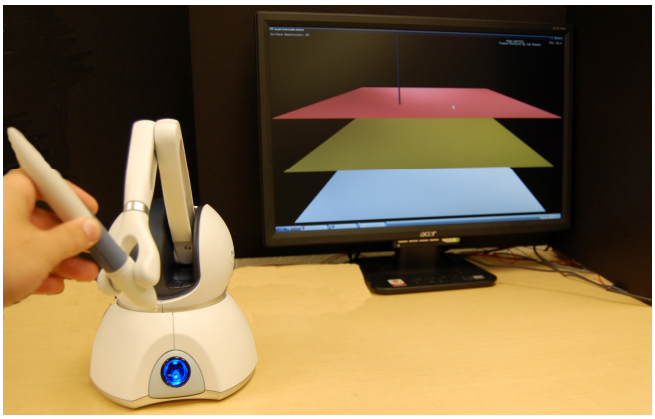

Fig. 4: Our test setup. The user controls the virtual needle and experiences forces by moving the Phantom Omni haptic device into the tissue.

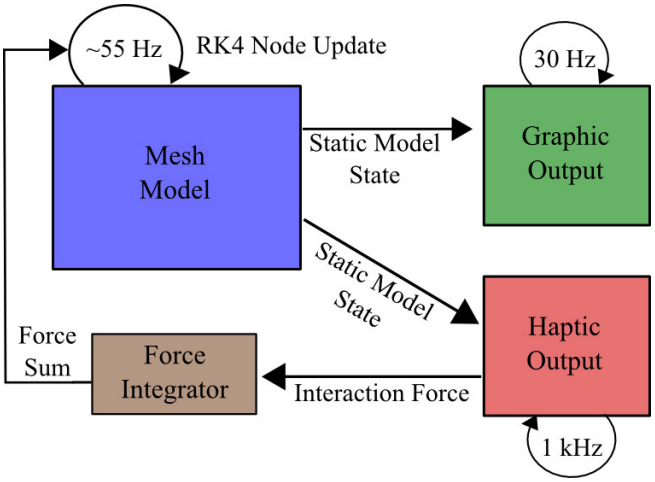

Fig. 5: Software architecture. The three mesh models respond to accumulated forces as soon as possible. The haptic force output computation uses the most recent static model state.

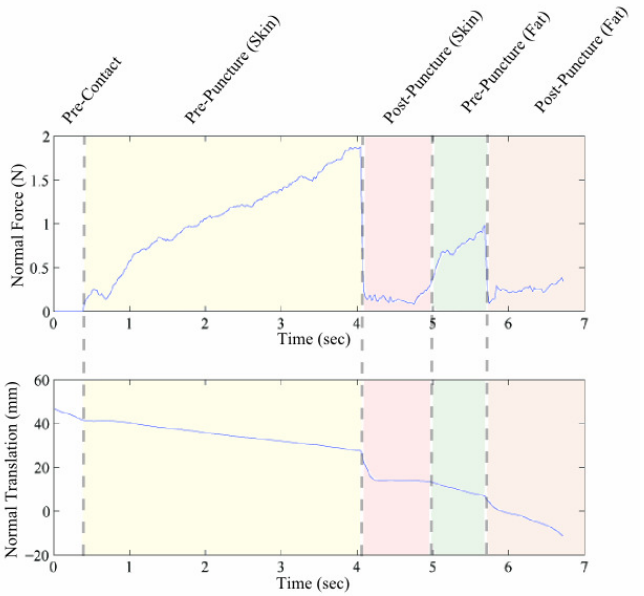

Fig. 6: A plot of the force experienced by the user and position over time for a puncturing trial. Notice the sharp drop-offs in force that coincide with surface puncture, and the lower Post-Puncture force levels caused by the viscous damping effects.

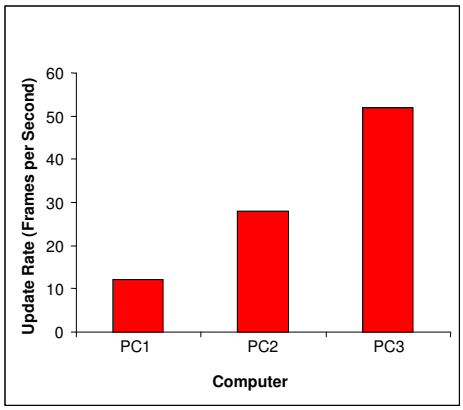

Fig. 7: The rate at which our model is able to update with three 25x25 node meshes on various computers. PC1 was a 1.4 GHz Pentium 4 with a NVIDIA Geforce3 graphics card. PC2 was a 2.4 GHz Pentium 4 with a NVIDIA Geforce4 MX graphics card. PC3 was a 2 GHz Quad Core with a NVIDIA 8800 GTS graphics card (only single core programming was performed).