*appeard in: Proceedings of Medicine Meets Virtual Reality 2002*

# Open Surgery Simulation

Daniel Bielser and Markus H. Gross

*Computer Science Department, ETH Zurich, Switzerland*
*email: {bielser, grossm}@inf.ethz.ch*

**Abstract** The design of simulators for surgical training and planning poses a great number of technical challenges. Therefore the focus of systems and algorithms was mostly on the more restricted minimal invasive surgery. This paper tackles the more general problem of open surgery and presents efficient solutions to several of the main difficulties. In addition to an improved collision detection scheme for computing interactions with even heavily moving tissue, a hierarchical system for the haptic rendering has been realized in order to reach the best performance of haptic feedback. A flexible way of modeling complex surgical tools out of simple basic components is proposed. In order to achieve a realistic and at the same time fast relaxation of the tissue, the approach of explicit finite elements has been substantially improved. We are able to demonstrate realistic simulations of interactive open surgery scenarios.

## 1. Introduction and Previous Work

The design of surgical simulators poses a great number of technical challenges: appropriate soft tissue models have to be chosen and the underlying differential equations have to be solved efficiently. Very often, surgical tools such as scalpels and hooks are involved, featuring complex interactions with the soft tissue. High speed haptic rendering of these tools requires sophisticated mechanical models.

Systems and algorithms for surgical training and planning proposed over the past years have mostly focused on minimal invasive surgery [1]. The underlying datastructures are often restricted to surface meshes [13] or regular grids [18], [9]. Apart from a few approaches, open surgery and real time soft tissue cutting is still largely unexplored. First approaches calculating intersection surfaces in retrospect can be found in the algorithms of [16] and [15]. The removing of entire tetrahedra in [5] is real-time but generates very uneven surfaces. A dynamic subdivision algorithm using an operator framework is introduced by the authors of this paper in [3] and has then be refined in [2] and [14] as well as extended to multiresolution grids by [8].

In this paper we will present interactive open surgery scenarios in which surgical hooks and large scalpel intersections are applied simultaneously. Our real-time system is based on arbitrary tetrahedral meshes and shows capable of interactively modeling incisions with high accuracy and topological freedom (chapter 3).

Previous volume collision detection algorithms like the one described in [3] and [2] have had difficulties in reliably registering all intersections between a tool and the tissue structures while the tissue is being deformed. We show a solution that solves the consistency problems even when the tissue is moving heavily (chapter 4).

The computationally most accurate but also most expensive methods for the modeling of elastic soft tissue use Finite Element procedures to solve the governing equations (e.g [11] and [12]). Recently, Boundary Element Methods (BEM), like [4] or [10] were proposed, condensing the solution into the domain boundary. Condensation, however, is very problematic because the whole stiffness-matrix has to be recomputed when cutting some surface elements. The formulation of explicit finite elements of [5], [7] and [6] helps here to reduce the computational complexity and allows for iteratively solving of the system. Whereas these works still solve the resulting differential equations with explicit numerical methods, the work in hand improves the approach of explicit finite elements by using the fast and robust (implicit) Theta-schemes, by parallelization, and by supporting individual time-steps for each node (chapter 5).

In most approaches, force feedback devices are utilized to implement the interface to the user. While haptical surface rendering of complex rigid objects has been already introduced in [17] and applied in many other works (e.g. [19]), we have developed algorithms to render surfaces of deforming objects including static and dynamic surface friction as well as more tool specific forms of tool-object-interaction. (chapter 2)

## 2. Simulation Overview

All main modules of a surgery simulator, like *Collision Detection*, *Geometrical Updates*, *Relaxation*, *Rendering* and *Haptics*, are realized as individual components of the simulation. During simulation the different components work in parallel. This enables the relaxation of the model in parallel with geometrical modifications and it makes the frame rate dependent only on the graphics hardware. In the following we focus on the interaction between the haptic simulation and the tissue model, which may be located on different machines connected via a TCP/IP connection.

The difficulty of haptic rendering deforming tissue is the slow response of the underlying physical model. In order to achieve the high update rates required for qualitatively sufficient haptic rendering we implemented a three stage simulation hierarchy (Fig. 1). On a haptic client the surface is approximated by a tangent plane and rendered with static and dynamic friction at an update rate of several thousand Hertz (**Haptics Simulation**). This plane is updated from the server side at high speed taking into account the actual local curvature of the model (**Haptics Geometry**). In a first step the output force is roughly approximated through the client by a simple spring model. It is then overlaid by the real external tissue forces as being computed by the relaxation process at the server side (**Relaxation**).

Fig. 2 explains the interplay of the three components in more detail: As soon as a tool approaches the surface an estimation of the tangent plane through which the tool may enter the surface is sent to the client. Once the first interaction of the tool with the tissue occurs (a) only the haptic client is fast enough to calculate a responding force $\mathbf{F}_{\mathbf{spring}}$ . A fraction of a second later (b), the reaction force from the tissue model $\mathbf{F}_{\mathbf{tissue}}$ arrives together with an

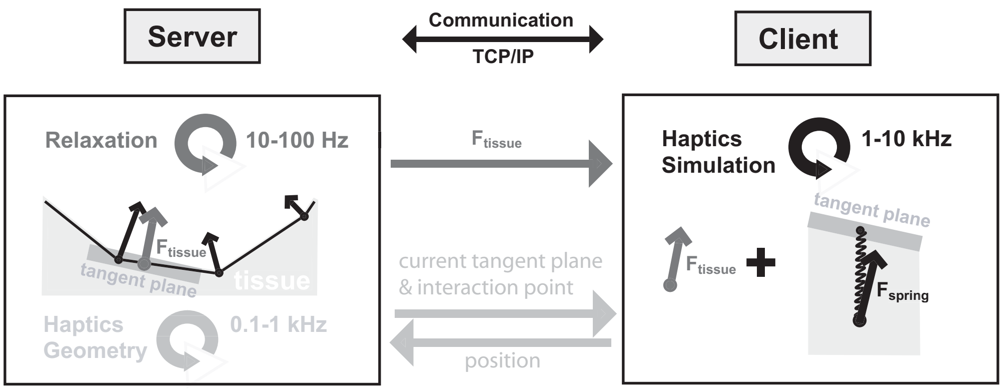

**Figure 1:** 3 stage simulation hierarchy

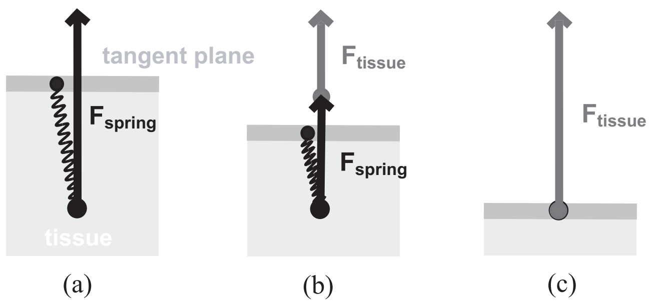

**Figure 2:** Interplay of the force components

update of the shifted tangent plane and is overlaid with $F_{spring}$ . If the tool does not move any further (c) $F_{spring}$ nearly vanishes and the haptic output consists solely of $F_{tissue}$ . With this procedure we manage to keep the point of the haptic tool outside the tissue.

## 3. Tool Tissue Interaction

As depicted in Fig. 3 the tools contain three representations of different complexity. The first representation is the graphical surface model consisting of several thousands of triangles. The second one is a simple line model used for collision detection. Finally, the third one presents the current interaction point for the haptic simulation.

The collision detection model (b) can be made up of any number of two different types of lines. One line-type results in deformation when interacting with an object and the other type cuts the tissue. For each line of the model the penetration with the object's surface is calculated. Special algorithms have been developed to handle the case of lines that have shown two or more intersections with the object's surface.

The haptic model (c), responsible for the calculation of the haptic output, is the same for all tools and supports only one interaction point. This is necessary because the applied PHANToM® device only supports positional force feedback for the tip of the pin. Therefore a procedure to determine the interaction point in relation to the current collision detection state has to be defined for each tool. The interaction point can be chosen from the colliding lines or computed as their average.

As for surgical tools, a *virtual scalpel* and a *surgical hook* have been realized so far. During an operation several of these tools can be activated. Depending on their orientation the hooks either deform the tissue while sliding over the surface or hook in and transmit the applied forces directly. In case of the scalpel the tetrahedra are cut progressively, thus continuously providing a consistent representation of the cut surface up to the actual position of the scalpel. Immediate computation of the interactions with the mesh structures allows us to simulate deformations resulting both from cutting and friction.

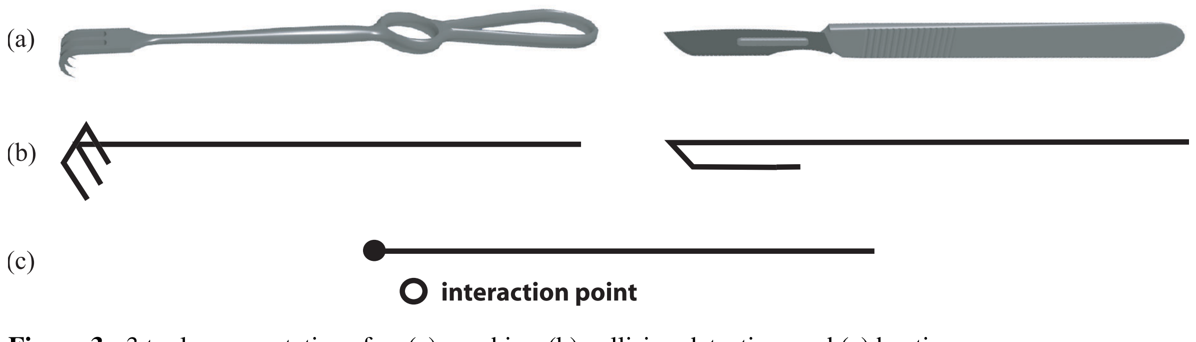

**Figure 3:** 3 tool representations for: (a) graphics, (b) collision detection, and (c) haptics

## 4. Local Collision Detection

Our system not only performs high performance surface proximity checks, but also supports local volume collision detection. Starting with some penetrated entry faces the algorithm iterates through the tetrahedra mesh along the current scalpel blade. With these penetrated tetrahedra as starting points, all tetrahedra affected by the preceded motion of the scalpel are retrieved by a recursive algorithm. In order to compute scalpel interactions even within heavily moving tissue structures, the local volume collision detection algorithms as described in [2] have been modified in the following way:

Each time the tetrahedra penetrated by the current scalpel blade are determined, the collision points between the tetrahedra faces and the blade line are calculated and stored locally inside the tetrahedra in form of barycentric coordinates (**ENPi** in Fig. 4 (a)). In contrast to storing a global scalpel position, the locally barycentric intersection points follow potential mesh transformations (Fig. 4 (b)). No matter how large the positional displacements of the mesh nodes are, each face intersection point **ENPi** stays relative to its face's nodes at the same position.

In order to actually mark the penetration between the scalpel trajectory and the tetrahedral edges a surface is spread between the previous marked collision points at time $t_i$ and the current position of the scalpel blade at $t_{i+1}$. From each line segment between two face intersection points (e.g. **ENP1** and **ENP2**) a triangle is spanned to one of the current scalpel intersection points **currentSP** or **currentENP**. From a topological point of view this procedure guarantees that the same tetrahedra edges will be cut whatever the motion of the tetrahedra nodes is.

Whereas in Fig. 4 only one tissue intersection is depicted, in real simulation situations several distinct parts of a tool may intersect the tissue at the same time. Approximating the tool by a set of lines, the two types of intersections symbolized in Fig. 5 (**ENP1** to **EXP3** and **ENP4** to **SP**) and the spaces in between need to be treated. The corresponding algorithm constructs the swept surface by proceeding from tetrahedron to tetrahedron and always constructing triangles between its entry point and the entry point of the subsequent tetrahedron. In order to bridge the gaps between two line segments the exit point of the last tetrahedron of a segment is always linked with a triangle with the entry point of the first tetrahedron of the next segment. The intersection points between the trajectory of the tool tip and the tetrahedra faces have to be calculated separately. For that reason a line is drawn between the current tool tip **currentSP** and its previous position **SP**, which both are stored as 4D barycentric coordinates relative to the nodes of the surrounding tetrahedron.

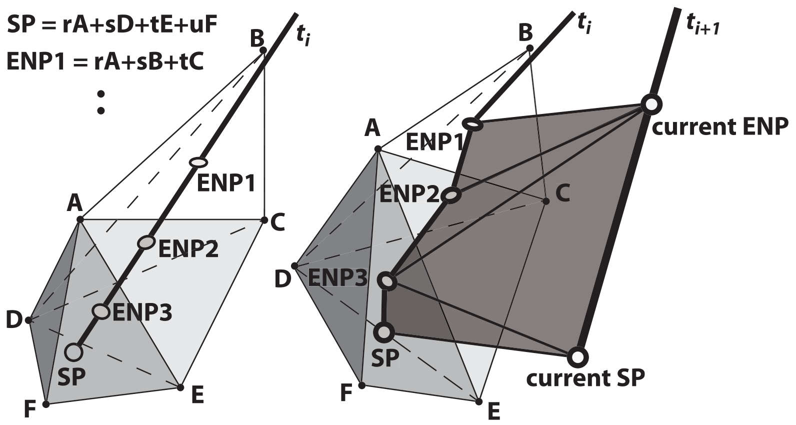

**Figure 4:** Benefit of local intersection points: (a) initial tetrahedra mesh (b) mesh after transformation

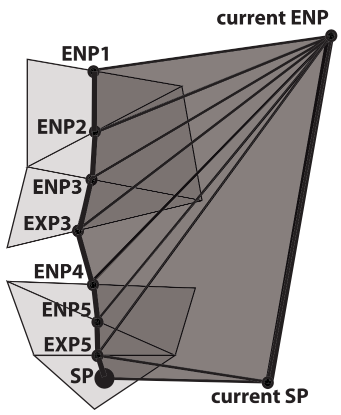

**Figure 5:** Collision algorithm

## 5. Relaxation

For the calculation of the underlying tissue structures two different physical models have been applied. On the one hand there exists already a very fast implicitly solved mass-spring system [2] and on the other hand we have developed an improved version of the more realistic tensor mass system introduced in [5]. This approach bases on continuum mechanics and describes the energy of elasticity in the tissue with linearized stresses as

$$E_{\text{elast}} = \int_V \left( \frac{\lambda}{2} \cdot (trE)^2 + \mu \cdot tr(E^2) \right) dV$$

whereas $tr$ is the trace of the matrix and $\lambda$, $\mu$ are the two Lame-coefficients.

$$\lambda = \frac{\nu \cdot E_{\text{Young}}}{(1+\nu) \cdot (1 - 2 \cdot \nu)} \qquad \mu = \frac{E_{\text{Young}}}{2 \cdot (1+\nu)} \qquad E_{\text{Young}}\text{: modulus of elasticity} \quad \nu\text{: poisson ratio}$$

The energy thereby is defined by the displacement vector $\vec{U}$ in relation to the restposition as

$$E = \frac{1}{2} \cdot (\nabla \vec{U} + (\nabla \vec{U})^T) \qquad \nabla\text{: gradient}$$

In order to achieve a local calculation scheme the nodes are visited iteratively and their positional changes are calculated by applying Newtons equation. The external force is obtained by precalculation of the force parts of the participating elements in the direct adjacency. The influence of the tissue force onto a specific node is divided into a part for the incident edges (edge tensors) and into one for the neighboring nodes (node tensors).

The indicated explicit finite element approach adapts very well to our parallelized iterative solving strategy which has already been successfully used to speed up mass-spring systems [2]. In contrast to the explicit numerical methods of the initial approach [5] we propose implicit Theta-schemes. This class of stable numerical methods enables to perform the time integration of the actually stiff differential equations with large time steps.

The $\theta$-scheme evaluates the partial derivation $dx/dt = f(x, t)$ two times, one time at the current position $x_i$ and one time at the subsequent position $x_{i+1}$. This two values are then weighted with a parameter $\theta$.

$$x_{i+1} = x_i + \Delta t \cdot ((1-\theta) \cdot f(x_i) + \theta \cdot f(x_{i+1}))$$

Furthermore, we support adaptive time steps for each individual node in order to better cope with the stiff differential equation system. For an absolute stable convergence of the relaxation process the size of the time step of each single node is controlled by fixing the largest possible velocity of a node. Whenever a node exceeds this upper bound its time-step will be halved and the node's position will be recalculated as long as the bound is not met.

## 6. Results

A prototype of a surgery simulator supporting all the features described above has successfully been realized. The system allows high quality rendering at real-time frame rates and reacts interactively for substantial mesh sizes (up to 3000 tetrahedra). Due to the progressive subdivision of the tetrahedral mesh, the virtual scalpel can be guided with high precision. The haptic output of the tool–tissue interaction for both of the implemented tools is free of any vibrations and oscillations and is considered realistic according to various test persons. The figures 6 to 10 give some sequential screen shots from the simulation of a operation in the abdominal area. The processed model consists of initially 3800 tetrahedra. The simulation setup is depicted in figure 11. (Visit the project website *http://graphics.ethz.ch/artist* in order to see the figures in color.)

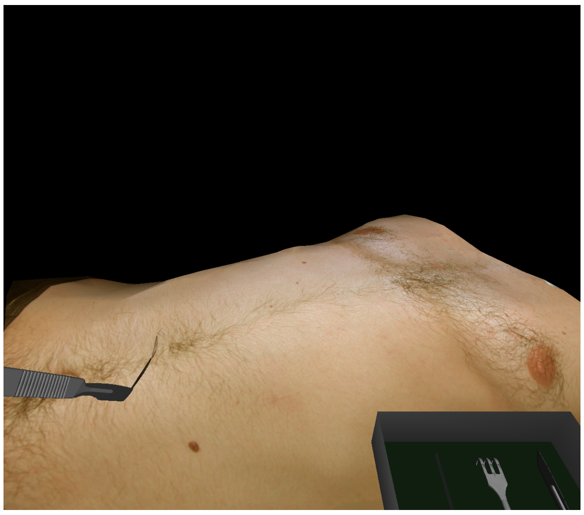
**Figure 6:** First scalpel cut in order to open the skin tissue

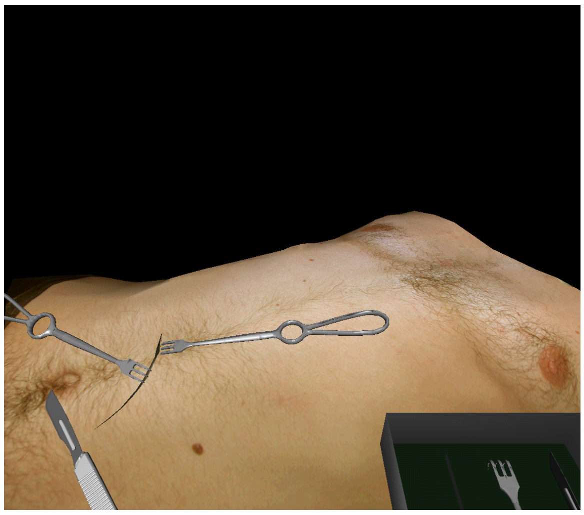
**Figure 7:** Insertion of the hooks

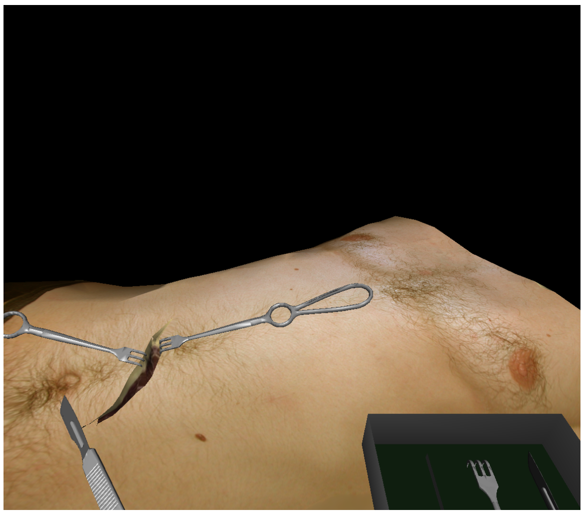
**Figure 8:** Opening the intersection

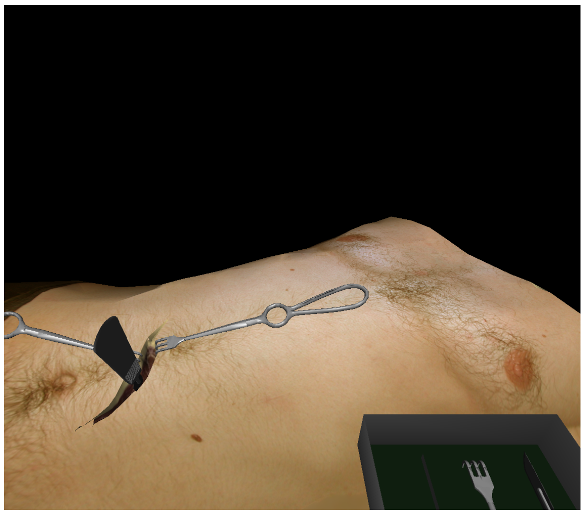
**Figure 9:** Fitting in the scalpel in order to perform a further cut

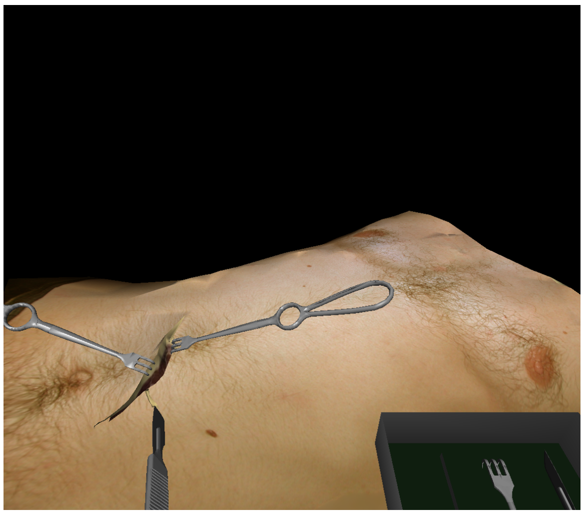
**Figure 10:** Carrying out a further cut

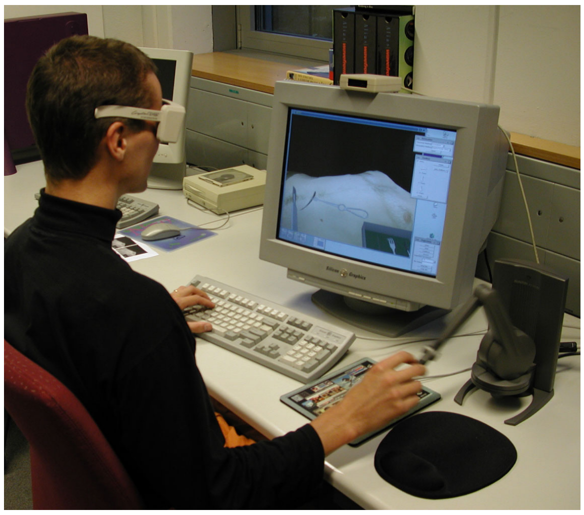
**Figure 11:** Simulation setup with PHANToM® force feedback device and stereo glasses

## 7. Conclusion and Future Work

We designed a realistic surgery trainer that features topological flexibility and accuracy for tool–tissue interactions occurring during open surgery procedures. The presented tools combined with an efficient and realistic tissue model allow us to simulate various surgical procedures with highly realistic force feedback.

Future work will expand the tool-tissue interaction approach of pure positional force feedback to the haptic rendering of torques by taking into account multiple interaction points with the tetrahedra mesh.

## 8. Acknowledgement

We would like to thank Remo Ziegler, Stefan Benkler, and Arno Jost for their contribution to the current implementation of our surgery training software.

## 9. References

[1] C. Basdogan, Chih-Hao, and M. A. Srinivasan. "Simulation of tissue cutting and bleeding for laparoscopic surgery using auxiliary surfaces." In *Medicine Meets Virtual Reality*, pages 38–44. IOS Press, 1999.

[2] D. Bielser and M. H. Gross. "Interactive simulation of surgical cut procedures." In *Proceedings of the Pacific Graphics 2000*, pages 116–125, 2000.

[3] D. Bielser, V. A. Maiwald, and M. H. Gross. "Interactive cuts through 3-dimensional soft tissue." In *Proceedings of the Eurographics '99*, volume 18, pages C31–C38, 1999.

[4] M. Bro-Nielsen and S. Cotin. "Real-time volumetric deformable models for surgery simulation using finite elements and condensation." *Computer Graphics Forum*, 15(3):C57–C66, C461, Sept. 1996.

[5] 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.

[6] G. Debunne, M. Desbrun, M.-P. Cani, and A. H. Barr. "Dynamic real-time deformations using space & time adaptive sampling." In *SIGGRAPH'01 Proceedings*, pages 31–36, 2001.

[7] G. Picinbono and H. Delingette and N. Ayache. "Non-linear and anisotropic elastic soft tissue models for medical simulation." In *ICRA2001: IEEE International Conference Robotics and Automation*, Seoul Korea, May 2001.

[8] F. Ganovelli, P. Cignoni, C. Montani, and R. Scopigno. "A multiresolution model for soft objects supporting interactive cuts and lacerations." In *Proceedings of the Eurographics 2000*, volume 19, pages C271–C282, 2000.

[9] S. F. Gibson. "3d chainmail: a fast algorithm for deforming volumetric objects." In *Proceedings 1997 Symposium on Interactive 3D Graphics*, pages 149–154, Apr. 1997.

[10] D. L. James and D. K. Pai. "Accurate real time deformable objects." In *SIGGRAPH Proceedings*, pages 65–72. ACM Press, 1999.

[11] R. Koch, M. Gross, D. von Bueren, G. Frankhauser, Y. Parish, and F. Carls. "Simulating facial surgery using finite element models." In *Proceedings of SIGGRAPH 96*, pages 421–428, 1996.

[12] R. Koch, S. Roth, M. Gross, A. Zimmermann, and H. Sailer. "A framework for facial surgery simulation." *Technical Report No. 327, Computer Science Department, ETH Zurich*, 1999.

[13] U. Kuehnapfel, C. Kuhn, M. Huebner, H. Krumm, H. Maafl, and B. Neisius. "The karlsruhe endoscopic surgery trainer as an example for virtual reality in medical education." In *Minimally Invasive Therapy and Allied Technologies*, volume 6, pages 122–125. Blackwell Science Ltd., 1997.

[14] A. B. Mor and T. Kanade. "Modifying soft tissue models: Progressive cutting with minimal new element creation." In *CVRMed-Proceedings*, volume 19, pages 598–607, 2000.

[15] K. D. Reinig, H. L. Pelster, V. M. Spitzer, T. B. Johnson, and T. J. Mahalik. "More real-time visually and haptic interaction with anatomical data." In K. M. et al, editor, *Medicine Meets Virtual Reality*, pages 155–158. IOS Press, 1997.

[16] K. D. Reinig, C. G. Rush, H. L. Pelster, V. M. Spitzer, and J. A. Heath. "Real-time visually and haptically accurate surgical simulation." In S. H. H. Sieburg and K. Morgan, editors, *Health Care in the Information Age*, pages 542–546. IOS Press, 1996.

[17] D. Ruspini, K. Kolarov, and O. Khatib. "The haptic display of complex graphical environments." In *SIGGRAPH Proceedings*, pages 345–352, 1997.

[18] N. Suzuki, A. Hattori, S. Kai, T. Ezumi, and A. Takatsu. "Surgical planning system for soft tissues using virtual reality." *Medicine Meets Virtual Reality*, pages 159–163, 1997.

[19] T. V. Thompson and E. Cohen. "Direct haptic rendering of complex trimmed nurbs models." In *Proceedings of Symposium on Haptic Interfaces*, 1999.