*Proc. HAPTICS 2002,* Orlando, FL, 2002, 66-72

# Haptic Volume Interaction with Anatomic Models at Sub-Voxel Resolution

A. Petersik[1], B. Pflesser[1], U. Tiede[1], K.H. Hoehne[1], R. Leuwer[2]
[1]Institute of Mathematics and Computer Science in Medicine (IMDM), [2]ENT-Clinic
University Hospital Hamburg-Eppendorf, Germany
petersik@uke.uni-hamburg.de

## Abstract

*In this paper, a new approach for haptic volume interaction with high resolution voxel-based anatomic models is presented.*

*The haptic rendering is based on a multi-point collision detection approach which provides realistic tool interaction with the models. Both haptics and graphics are rendered at sub-voxel resolution, which leads to a high level of detail and enables the exploration of the models at any scale.*

*Forces are calculated at an update rate of 6000 Hz and sent to a 3-Degree-of-Freedom (3-DOF) force-feedback device. Compared to point-based haptic rendering, the unique approach of the multi-point collision detection in combination with sub-voxel rendering provides more realistic and very detailed haptic sensations.*

*As a main application, a simulator for petrous bone surgery was developed. With a simulated drill, bony structure can be removed and the access path to the middle ear can be studied.*

## 1. Introduction

In medical applications it is often not sufficient to only have a graphical representation of the anatomy. Especially the sense of touch is of value for a range of applications. In contrast to our other senses it allows us to simultaneously explore and interact with our environment. This is intensely used in surgery.

The recent development of high quality force-feedback devices like the Phantom (Sensable Technologies Inc.) has brought the sense of touch to computer applications. With such devices it is possible to integrate force-feedback in surgical simulators and other medical applications. However, applications in surgery simulation have to overcome conflicting requirements of complexity and accuracy of the anatomical model and the speed of interaction with the model. Most of these applications concentrate on the simulation of elastic deformations of soft tissue. In contrast to that the simulation of material removal in medical applications is a less developed field and simulation systems either do not include cutting operations at all, or in a simplified manner, which do not provide the 'look and feel' close to a real incision.

Moreover haptic rendering is mostly based on traditional computer graphics methods where objects are represented by polygons only. However, using a surface based representation to create models for medical applications, the knowledge about the interior structures of the organs is lost. This knowledge is very important both for the simulation of interactive cutting operations such as required for surgery simulations and for the display of cuts which is essential for medical applications. Also collision detection in complex environments is easier with a volume based representation.

In addition we realized that today 3-DOF haptic rendering is mostly point-based, i.e. only one point is used to calculate collisions and forces. This induces several problems:

- Discontinuities (e.g. sharp edges) on the surface can lead to discontinuities in the haptic display.
- The virtual tool can reach points which can not be reached by the simulated real world tool. (A large drill could enter a small hole.)

One goal of the work presented in this paper is to develop a haptic rendering algorithm which is able to display arbitrary complex anatomic models with high realism and accuracy. The problems pointed out above inhibit a realistic impression of the virtual models and thus must be eliminated.

Another goal was to not only enable the haptic exploration of the anatomic models but also to be able to modify those models interactively with simulated real world tools. We implemented a petrous bone surgery simulator, where a sphere-shaped tool which simulates a drill can be used both to explore and to manipulate the anatomic model.

## 2. Related work

As in computer graphics, most haptic rendering approaches are using triangle-based data representations. Since high-detailed models would lead to a large number of triangles and volume modification needs knowledge about the interior structure of the objects, a voxel-based model is more suitable for volume interaction with anatomic models.

In the early days of haptic rendering mostly penalty based methods were used [5]. These methods use precomputed force fields or forces that are computed based on the shortest distance to the object surface only. A problem of this approach is that there are points in an object which have the same distance to several surface points, e.g. a sphere's center point has the same distance to all surface points. By crossing such points with the haptic device the force direction changes. The result is that the user can press easily through objects. This is a severe problem, especially when working with high detailed models and small structures.

Later constraint-based approaches were introduced by [11, 9]. These approaches use an intermediate object which never penetrates an object in the environment. The intermediate object (called God-object or Proxy) remains on the surface. The force which is applied to the haptic device is proportional to the difference vector between physical device position and proxy object. The haptic rendering algorithm has to update the proxy position in respect to the device position by locally minimizing the distance to the device position. Since these calculations have to be performed on-the-fly, contraint-based approaches are computationally more expensive than penalty-based approaches.

An early approach of haptic interaction in volume visualization was presented in [1]. While this approach allows exploration and modification of the volumetric data, the focus of this approach was not to give the user a realistic haptic feedback, but to give a feedback which is helpful for exploration and modification purposes. Furthermore this approach uses a single-point contact model and a penalty-based approach which does not fulfill our needs for a realistic interaction.

Recently, multi-point collision detection approaches were developed. With these approaches, more realistic simulations of tools and tool interaction can be achieved.

In a single-point collision detection approach one could e.g. enter a small hole with a large tool, which is not realistic. Of course multi-point collision detection is computationally more expensive, but when realistic tool interaction is required, single-point models are not sufficient.

The voxel-based approach to haptic rendering presented in [6] enables 6-DOF manipulation of a modestly sized rigid object within an arbitrary complex environment of static objects. This approach provides a realistic haptic feedback in regard to the exact shape of the tool. One drawback of this method is that the haptic rendering is at voxel level accuracy only which is not sufficient for medical applications.

In [7] an algorithm for volume cutting was presented. This approach is working at sub-voxel resolution, and therefore provides the data for a detailed rendering.

## 3. Methods

To develop a simulator for petrous bone surgery the following three points had to be considered:

- Haptic rendering should be based on a multi-point collision detection to allow a realistic tool-object interaction.
- For volume interaction an algorithm is needed which works with sub-voxel precision to be able to simulate small tools as they are used in petrous bone surgery.
- To get haptic feedback while drilling into the bone a suitable haptic rendering algorithm is needed.

### 3.1 Data representation

The volume data of our anatomic models is represented by attributed voxels (volume elements) which usually have a size of $1mm^3$. The attributes describe data like RGB or density values and associated organs. The associated organ is determined during the segmentation process which is mainly done by applying thresholds to the different structures. If segmentation is not possible by thresholding due to low density changes at object borders, segmentation can also be done by manual painting on slices.

Since our voxel-based representation does not contain an explicit representation of the object surfaces, the surfaces must be calculated based on the segmentation data. This is done by a ray-casting algorithm [10] which renders iso-surfaces at sub-voxel resolution based on the partial volume effect and density value of the voxels. Such calculations are computationally expensive and thus they should be minimized to the extend absolutely necessary. On the other hand our sub-voxel approach leads to very detailed surfaces which can be explored at any scale.

The same ray-casting algorithm is used for both graphic and haptic rendering, and thus leads to a congruent and high detailed graphic and haptic display.

### 3.2 Tool representation

For multi-point collision detection the tool is represented by a number of sample points which are distributed at preferably equal distances over the tool surface. Each of these points is checked whether it collides with the objects or not. The distance of two neighbored tool surface points

must be smaller than the diameter of the smallest object in the scene. Otherwise such small objects can not be explored in a realistic manner. Additionally every point has an associated normal vector which is pointing to the inside of the tool (Fig. 1). All inward pointing surface normals must have the same length.

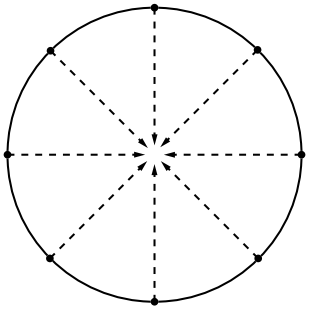

**Figure 1. Tool representation by surface points and inward pointing normal vectors**

The inward pointing vectors are used to find the static object's surface. An advantage of this approach is that no computation of surface normals for the static object is required.

In our implementation of the petrous bone surgery simulator we are using a sphere-shaped tool, which simulates a drill. To get an adequate representation of the tool shape while reducing computation to a minimum, we used 26 sample points on the sphere's surface.

Of course the tool representation could be extended to more general shapes. An algorithm for determination of tool surface points was presented in [8].

### 3.3 Haptic surface rendering

To get a realistic haptic surface rendering while not manipulating the volume we decided to use a multi-point collision detection approach. This provides a more realistic haptic rendering compared to a single-point collision detection approach. Additionally the problems that occur when probing a drilled hole (fig. 2) with a single-point collision detection algorithm are eliminated.

Our haptic device gives us the cartesian coordinates of the tool with a selectable update rate between 1000Hz and 10000Hz. The higher the update rate is the higher is the maximum stiffness which can be haptically rendered. To simulate the presence of virtual objects, we have to check for a collision at every iteration step. When a collision occurs, we must send a force to the device, which pushes the user back to a collision-free location on the object's surface.

Thus to get a realistic rendering of the surfaces, the following two parameters must be computed whenever a collision between tool and static objects occurs:

- Direction and
- Magnitude of collision force

Both variables must be computed with high precision to allow a realistic interaction for e.g. petrous bone surgery.

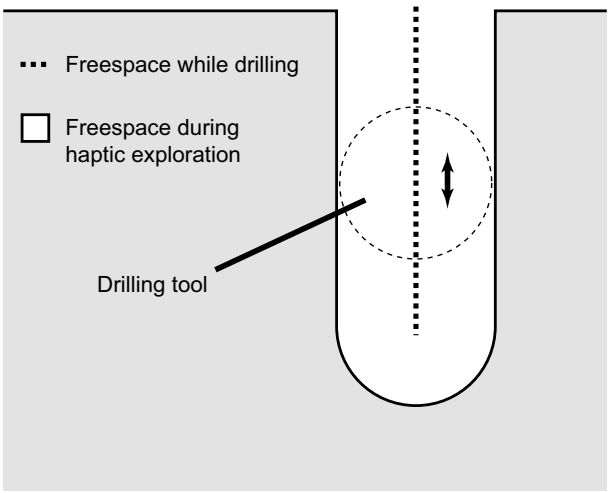


**Figure 2. Problems while modifying the volume with single-point collision detection**

Our multi-point collision detection algorithm was inspired by the work described in [6]. However our approach differs from this work in several points. While our model is also using a voxel representation for the static objects, the exact location of the surfaces is calculated by a ray-casting algorithm at sub-voxel resolution (see 3.1). This leads to a more precise calculation of both force direction and magnitude. The algorithms presented in [6, 8] can not provide the precision which is needed in our cases, since the static objects are voxelized in a binary manner. Another problem of the algorithm presented in [6] is that force incontinuities can appear when the tool crosses voxel bounderies of the static objects under sliding motion. This problem was addressed in [8], but since we are using an isosurface representation we do not have to deal with this problem. With our approach the precision is not limited by the size of the voxels.

Another improvement we have made is the representation of the dynamic object. While the dynamic object in [6] is voxelized and the center points of the voxels are used for the collision detection, we are using sample points which are located exactly on the surface of the dynamic object.

A general algorithm which is doing that was presented in [8]. Since we are working on a petrous bone surgery simulator which on the one hand requires high precision, but on the other hand does not involve a large amount of different tools, we decided to determine the tool's surface point manually with regard to the precision. Therefore it is important that the points are distributed at preferably equal distances over the tool surface.

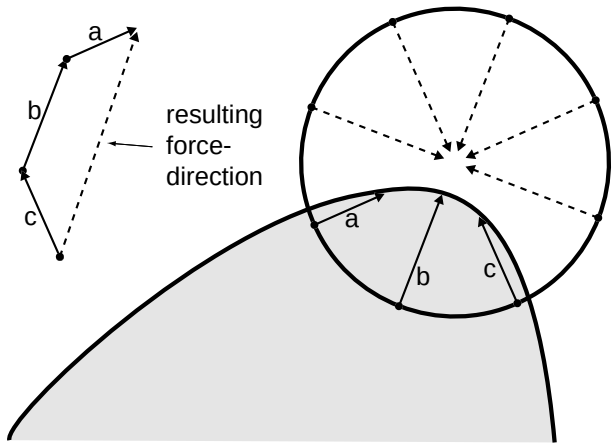


**Figure 3. Calculation of force direction at the object surface by adding tool vectors inside the object**

To calculate the collision force direction, the surface points of the tool are checked, whether they are inside or outside the object. All surface points which are inside the volume are traced along the inward pointing normal until the surface is found (Fig. 3, solid vectors) or the end of the normal is reached (Fig. 3, dotted vectors). All found vectors are added and the direction of the sum vector is the direction of the force vector which must be applied to the haptic device. As stated in [6], the summation of the found vectors would lead to force instabilities. As more of the tool's contact points collide with the static object, more stiffness is the result which creates a less stable contact situation like shown in figure 4, situation 2 where the sum generates a much too big force. The averaging of the found vectors on the other hand would lead to a stable contact situation but especially when opposite tool points have contact (occur in clefts, fig. 4 situation 3) the resulting force would be much too low. Even the sum leads to a force which is to small (fig. 4 situation 3).

Since forces which are much too big (like in fig. 4 situation 2) lead to instabilities we decided not to use force summing but to use a kind of averaging instead. To reduce the problems for opposite tool contacts while averaging, we

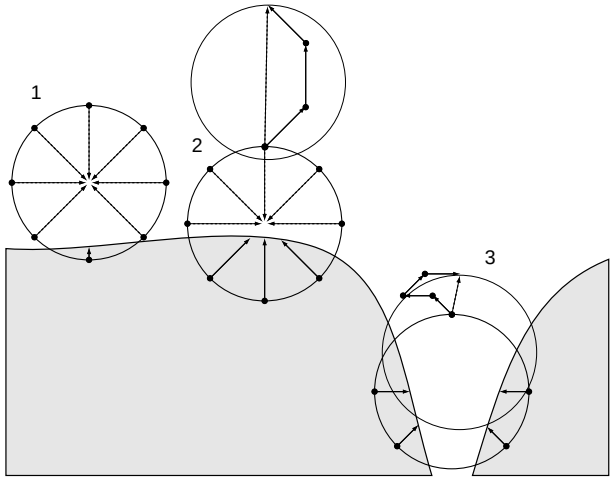


**Figure 4. Problems with force sum**

calculate the result force the following way:

$$force = \frac{forceSum}{(contacts \cdot 0.65 + 0.35)} \quad (1)$$

This approach is better than the force sum or the pure average as can be seen in fig. 4.

### 3.4 Proxy object algorithm

Any haptic device has a limited maximum force which can be applied to it. When the user pushes harder than this limit, it is possible that the physical position of the tool immerses completely into the virtual model. In this case the surface of the static object can not be found, because the majority of the inward pointing normals are in the object itself. Thus calculation of the force direction as described above is not possible anymore.

To overcome that limitation a modified proxy object algorithm [11, 9] was implemented. The idea behind a proxy object is to represent the device position by a virtual object which never penetrates objects. In free space, the position of the proxy object and of the device is identical. When the haptic device moves into an object, the proxy remains on the object's surface. When the proxy position is known, the resulting force vector is proportional to the difference vector between the proxy and the device position.

In order to update the proxy position on the static object's surface while the device is moving, the distance between the device position and the proxy must be locally minimized by regarding the surface constraints. Since searching for the local minimum would be computationally too expensive in our model, a simplified algorithm was implemented. Whenever more than a certain number of surface sample points of the dynamic object are in contact with an object (Object 2

in fig. 5), the way between proxy and new tool position is traced until the object surface is found. Starting from the surface position the way back to the proxy is traced until the number of contacts is below the limit (Object 3 in fig. 5) or until the proxy is reached. At that location the force vector can be calculated as described above and the proxy is set to that position (Object 4 in fig. 5). Since the difference between two successive positions is very small, this approximation gives a realistic feeling of the virtual objects.

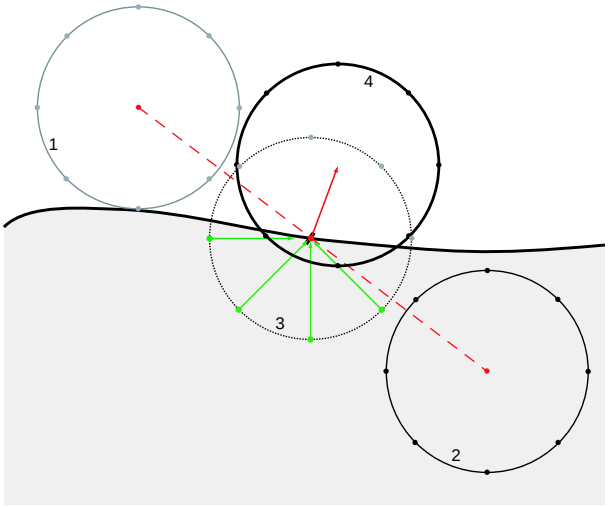


**Figure 5. The modified proxy algorithm**

### 3.5 Volume interaction

The sense of touch allows not only the exploration of the anatomy but also the interactive modification of the objects we are touching. In petrous bone surgery the drill allows the surgeon to feel and to drill the bone. In our simulator drilling is performed while pressing a button on the force-feedback device. Otherwise the structure of the petrous bone can be explored.

Our freeform volume modification algorithm which is descibed in detail in [7] is working with sub-voxel precision. The cutted parts are modeled by simulating the partial volume effect. This produces realistic structures even when using very small tools.

A drawback of this approach is that it consumes quite a lot of processing time, especially for the graphical rendering of the modified region. To compensate this our algorithm calls the volume modification routine asynchronously while the haptic process is running at constantly 6000Hz. In mean we achieve an update rate of 8Hz for the drilled structures. Since drill movements in petrous bone surgery are rather slow, this limitation still provides a realistic interaction.

### 3.6 Calculation of drilling forces

While the volume is modified, the calculation of the surface location can not be performed as described in chapter 3.3 and 3.4. Both simulation of the partial volume effect, which must be done to modify the volume with sub-voxel precision, and display update for the modified region consume too much processing time. Thus drilling forces must be calculated by a simplified algorithm. As a first approximation we apply a force which is opposite to the drilling direction. This can not be done by directly applying a force which is proportional to the distance between the current and the last device position, since this would lead to hard vibrations even at low drilling forces.

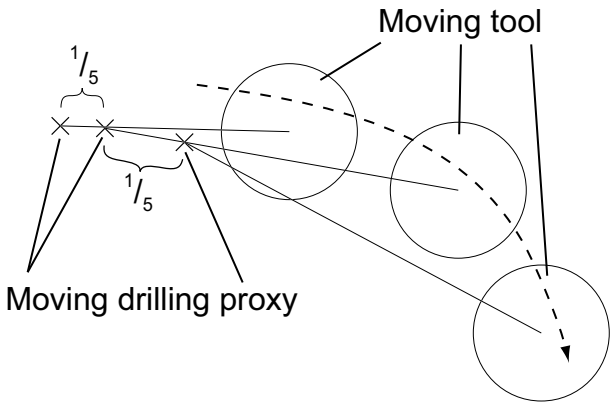


**Figure 6. The drilling proxy**

To overcome that problem, we developed a proxy for drilling. The position of the drilling proxy is updated at a lower frequency than the haptic update rate. In our case we use an update rate of 24Hz which is high enough not to be sensed by the user. At every haptic update, a force is applied to the haptic device which is propotional to the distance to the drilling proxy. The position of the drilling proxy is updated in the following manner:

- When the tool is in free space, the drilling proxy is set to the current device position
- While drilling, the proxy is set to a position on the connecting line between the last position of the drilling proxy and the current device position.

The more the new position of the drilling proxy is located near the old drilling proxy, the harder can be the maximum drilling force. A disadvantage of setting the new position very close to the old position is that under fast changing moving directions a feeling of inertia can be sensed. A good compromise for bone drilling is to set the drilling proxy to

the point of the old drilling proxy plus one fifth of the distance between old position and current device position. The applied force is 800N/m multiplied by the distance between the drilling proxy and the current device position.

## 4. Implementation

The petrous bone surgery simulator was integrated in the VOXEL-MAN [3] system, which provides the anatomic model, high-quality visual rendering and also freeform volume modification. A requirement of the system is a UNIX compatible operating system and, depending on the size of the anatomic model, sufficient main memory.

To fulfill these requirements, the system is implemented on a Compaq SP750 workstation. The workstation has two Pentium III Xeon processors running at 866 MHz and is equipped with 2GB RAM. As haptic device we are using a 3-DOF Phantom Premium 1.0A (Sensable Technologies Inc.). Our system is running SuSE Linux 7.1. For connecting the device directly to the system, we are using the open-source PHANToM Linux-driver (http://decibel.fi.muni.cz/phantom). To enable stereoscopic viewing with shutter glasses we are using a Windows computer with an Nvidia Quadro graphics board and ReflectionX (WRQ Inc.) XWindow software.

The haptic process runs at an update rate of 6000Hz. The most time consuming calculation is the calculation whether a point is in contact with an object or not. By using 26 sample points on the sphere, 60 such calculations are executed at every haptic update on average.

## 5. Results

With our approach we achieve a non-deformable high quality haptic rendering of arbitrary complex anatomic models [4] as shown in figure 7. The perception of spatial relationships is greatly enhanced with haptic feedback. Even very small surface details or small objects like nerves and vessels can be sensed realistically due to our sub-voxel based approach.

The main focus of our work was the interaction with non-deformable material (e.g. bone). With the algorithms presented in this paper we developed a system for the simulation of petrous bone surgery (fig. 8) in cooperation with the ENT-surgeon Dr. Leuwer, who has carried out more than thousand petrous bone surgeries. In the simulator, the haptic device is used to simulate drilling into the mastoid bone. Using a tool-based collision detection approach proved to be much more realistic than using point-based haptic interaction. Drilled holes and tubes in the bone could be intuitively probed with the haptic device and have the size as expected from the visual representation.

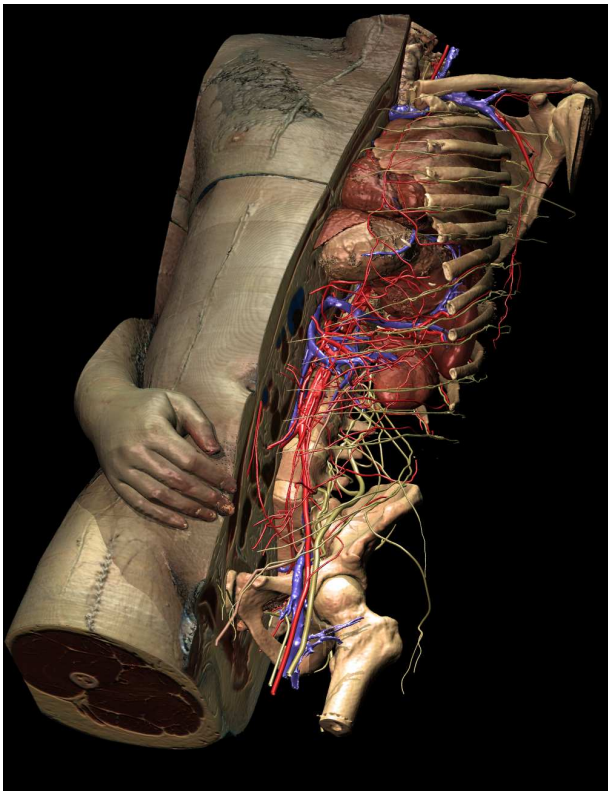

**Figure 7. Complex anatomical model. The algorithm can be used for its exploration.**

## 6. Conclusions and future work

In this paper we have presented an approach for realistic haptic rendering and volume interaction with anatomic models. It overcomes several problems of point-based haptic rendering. The shape based collision detection approach makes interactions more realistic, especially for the use in surgery simulations. Another important advantage is that surface details can be sensed as expected from the graphical representation. Sub-voxel rendering leads to both a graphical and haptic realistic, high detailed and congruent display. The volume interaction uses a sub-voxel based algorithm which allows the use of small tools, like drills used in petrous bone surgery.

While in the petrous bone surgery simulator only sphere-shaped tools are realized, future implementations will extend this to more general shapes. The calculation of the collision force magnitude will be investigated further to improve haptic rendering at locations with many tool object intersections as they appear especially in deep clefts.

Other improvements will be done to speed up graphical rendering of the modified structures. Our current imple-

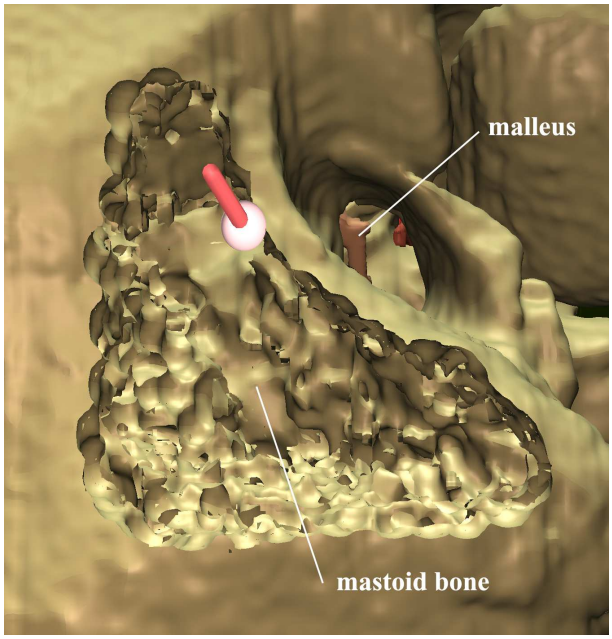


**Figure 8. Simulation of petrous bone surgery. The patient is in horizontal position, the malleus may be seen through the ear hole.**

mentation redraws the whole tool region while future implementation could reduce redrawing to the modified voxels only. Also ray-clipping in direction of the z-axis during redraw could speed up the process.

Calculation of the drilling forces will be investigated further to improve the sensation of the structures while drilling.

## References

[1] R. S. Avila and L. M. Sobierajski. A haptic interaction method for volume visualization. *Proceedings of Visualization 96*, pages 197–204, 1996.

[2] K. H. Höhne, M. Bomans, A. Pommert, M. Riemer, C. Schiers, U. Tiede, and G. Wiebecke. 3D-visualization of tomographic volume data using the generalized voxel-model. *Visual Comput.*, 6(1):28–36, 1990.

[3] K. H. Höhne, B. Pflesser, A. Pommert, M. Riemer, T. Schiemann, R. Schubert, and U. Tiede. A new representation of knowledge concerning human anatomy and function. *Nat. Med.*, 1(6):506–511, 1995.

[4] K. H. Höhne, B. Pflesser, A. Pommert, M. Riemer, R. Schubert, T. Schiemann, U. Tiede, and U. Schumacher. A realistic model of human structure from the Visible Human data. *Meth. Inform. Med.*, 40(2):83–89, 2001.

[5] T. M. Massie and J. K. Salisbury. The phantom haptic interface: A device for probing virtual objects. *ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems 1994*, 1:295–301, 1994.

[6] W. A. e. a. McNeely. Six degree-of-freedom haptic rendering using voxel sampling. *Computer Graphics (SIGGRAPH99 Proceedings)*, pages 401–408, 1999.

[7] B. Pflesser, U. Tiede, K. H. Höhne, and R. Leuwer. Volume based planning and rehearsal of surgical interventions. In H. U. Lemke, M. W. Vannier, K. Inamura, A. G. Farman, and K. Doi, editors, *Computer Assisted Radiology and Surgery, Proc. CARS 2000*, volume 1214 of *Excerpta Medica International Congress Series*, pages 607–612. Elsevier, Amsterdam, 2000.

[8] M. Renz, C. Preusche, M. Potke, H.-P. Kriegel, and G. Hirzinger. Stable haptic interaction with virtual environments using an adapted voxmap-pointshell algorithm. *Proc. of the Eurohaptics Conference*, pages 149–154, 2001.

[9] D. C. Ruspini, K. Kolarov, and O. Khatib. The haptic display of complex graphical environments. *Proc. of ACM SIGGRAPH*, pages 345–352, 1997.

[10] U. Tiede, T. Schiemann, and K. H. Höhne. High quality rendering of attributed volume data. In D. Ebert, H. Hagen, and H. Rushmeier, editors, *Proc. IEEE Visualization '98*, pages 255–262. IEEE Computer Society Press, Los Alamitos, CA, 1998.

[11] C. Zilles and K. Salisbury. A constraint-based god object method for haptics display. *Proceedings of IEEE/RSJ*, 1995.