Project Number	**IST-1999-12175**
Project Title	**IERAPSI**

Deliverable Number	**D4.2**
Deliverable Title	**Petrous bone surgical simulation platform**
Version Number	**1.0**
Nature	**Report**
Dissemination Level	**Internal**
Contractual Date of Delivery	**PM 36**
Actual Date of Delivery	**PM 36**
Primary Authors	**Enrico Gobbetti** **CRS4** **Gianluigi Zanetti** **CRS4**
Other Contributors	**Marco Agus** **CRS4** **Andrea Giachetti** **CRS4** **Antonio Zorcolo** **CRS4**

Contents

Executive Abstract **v**

Synopsis **1**

1 General background and context **3**

2 Surgical simulation software kernel architecture **7**
2.1 The decoupled simulation model 7
2.2 Burr-bone interaction and haptic feedback 9
2.2.1 Multi-scale spatial description 11
2.2.2 Multi-scale erosion 12
2.2.3 Other contributions to the haptic response 13
2.3 Secondary visual effects 13
2.3.1 Sample–Estimate-Hold Interface 15
2.4 Real–time visual rendering 15
2.4.1 Shaded direct volume rendering of dynamic volumes 16
2.4.2 Reducing fill-rate bottleneck 18

3 Hardware System Configuration **19**

4 System Evaluation **21**
4.1 General system performance 21
4.1.1 Multi-resolution Force Evaluation 21
4.1.2 Visual feedback 23
4.2 Test sessions 23

References **25**

Executive Abstract

This Report has been prepared in fulfilment of **Deliverable D4.2**, required as a result of Work Package 4 (*Real time physically based surgical simulators*) of the EU Framework V Project ***IERAPSI***, *An Integrated Environment for the Rehearsal and Planning of Surgical Interventions* (IST-1999-12175).

Deliverable **D4.2** relates to the **Petrous bone surgical simulation platform**, the second and last of the two main expected results of Work Package 4.

The present document provides a technical description of the software system produced. The document is divided in the following parts:

- Section 1 provides general background information on the functional and implementation specification. This section summarizes the findings reported in deliverable D2, "Surgical Procedures and Implementation Specification", relevant to the development of the "Petrous bone surgical simulation platform", deliverable D4.2.

- Section 2 provides a description of the surgical simulation software kernel architecture and its components. In particular, details will be provided on the techniques used to: simulate the interaction between virtual surgical tools and the bone tissue; simulate the generation of obscuring effects, due, e.g., to the accumulation of bone dust; visualize the simulated physical system at frame rates compatible with real-time interaction with the system.

- Section 3 provides a description of the hardware system configuration used to test the surgical simulation software kernel architecture. The hardware system configuration used includes: a single-processor PIV/1400 MHz with 256 MB PC133 RAM for to control the haptic devices; a dual-processor PIII/800 MHz with 512 MB PC800 RAM and a NVIDIA GeForce 4 Ti 4600 running a Linux 2.4 kernel for the simulation; a Phantom Desktop haptic device for the dominant hand; a Phantom 1.0 haptic device for the non-dominant hand; an n-vision VB30 binocular display for presenting images to the user.

- Section 4 summarizes the results of the system technical and clinical evaluation. The main finding is that the petrous bone surgical simulation platform respects

the requirements listed in D2 and enables realistic simulation of burring procedures. The tests have been carried out in collaboration with surgeons and trainees from the University of Pisa.

The report concludes with a bibliography of cited reference work. An accompanying video (available on the deliverable CD-ROM) further illustrates the petrous bone surgical simulation platform with live sequences comparing a real and a simulated surgical procedure performed on the temporal bone.

Copyright Notice

Synopsis

Purpose of the Document

This Report has been prepared in fulfilment of **Deliverable D4.2**, required as a result of Work Package 4 (*Real time physically based surgical simulators*) of the EU Framework V Project ***IERAPSI***, *An Integrated Environment for the Rehearsal and Planning of Surgical Interventions* (IST-1999-12175).

Deliverable **D4.2** relates to the **Petrous bone surgical simulation platform**, the second and last of the two main expected results of Work Package 4.

The present document provides a technical description of the software system produced.

Structure of the Document

- Section 1 provides general background information on the functional and implementation specification. This section summarizes the findings reported in deliverable D2, "Surgical Procedures and Implementation Specification", relevant to the development of the "Petrous bone surgical simulation platform", deliverable D4.2.

- Section 2 provides a description of the surgical simulation software kernel architecture and its components. In particular, details will be provided on the techniques used to: simulate the interaction between virtual surgical tools and the bone tissue; simulate the generation of obscuring effects, due, e.g., to the accumulation of bone dust; visualize the simulated physical system at frame rates compatible with real-time interaction with the system.

- Section 3 provides a description of the hardware system configuration used to test the surgical simulation software kernel architecture. The hardware system configuration used includes: a single-processor PIV/1400 MHz with 256 MB PC133 RAM for to control the haptic devices; a dual-processor PIII/800 MHz with 512 MB PC800 RAM and a NVIDIA GeForce 4 Ti 4600 running a Linux 2.4 kernel for the simulation; a Phantom Desktop haptic device for the dominant

hand; a Phantom 1.0 haptic device for the non-dominant hand; an n-vision VB30 binocular display for presenting images to the user.

- Section 4 summarizes the results of the system technical and clinical evaluation. The main finding is that the petrous bone surgical simulation platform respects the requirements listed in D2 and enables realistic simulation of burring procedures. The tests have been carried out in collaboration with surgeons and trainees from

The report concludes with a bibliography of cited reference work.

An accompanying video (available on the deliverable CD-ROM) further illustrates the petrous bone surgical simulation platform with live sequences comparing a real and a simulated surgical procedure performed on the temporal bone.

Chapter 1

General background and context

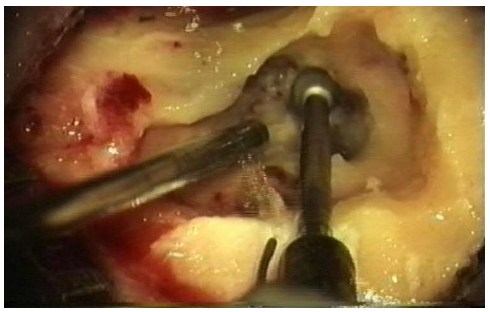
A detailed task analysis, following ISO 13407 [ISO99], has been carried out in order to identify the essential ergonomic components [Sto01, AGG$^+$02a]. The analysis involved a review of existing documentation, training aids, and video recordings, interviews with experienced operators, as well as direct observation of the procedure being performed in theater. This analysis, and its implication to the functional requirements of the system, have been extensively discussed in, respectively, deliverable D2 part 1 *Human Factor Analysis*, [Sto01] and D2 part 2 *Surgical Simulation Subsystem Requirements and Functional Specification*, [GZ01]. In this chapter we briefly summarize the principal findings reported in those documents.

In the typical mastoidectomy surgical setup, Fig. 1.1(a), the ENT surgeon looks at the region of interest through a stereoscopic microscope and holds in his hands a high speed burr and a sucker. These tools are used, respectively, to cut the bone and to remove water (used to cool the burr bit) and bone paste produced by the mixing of bone dust with water.

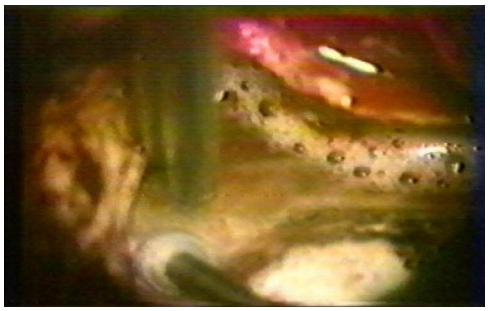
Subjective analysis of video records, together with *in-situ* observations highlighted a correlation between drilling behaviours and type and depth of bone. In the case of initial cortex burring and recess preparation for, e.g., a cochlea implant receiver/stimulator, drill tip/burr motions of around 0.8 cm together with sweeps over 2-4 cm were evident, as were fine flexion and extension movements of the forefinger and thumb around the drill. Shorter (1-2 cm) motions with rapid lateral strokes characterized the post-cortex mastoidectomy. For deeper drilling, $\sim$1 cm,- strokes down to 1 or 2 mm were evident with more of a "polishing" motion quality, guided using the contours from prior drill procedures. "Static" drill handling was also noted, eroding bone tissue whilst maintaining minimal surface pressure.

As for the visual effect of the drill on the surface of the bone, the task analysis hilighted that the graphical process must simulate drill site obscuration by bone dust paste, because its absence would reduce the importance placed by a trainee on the need for regular irrigation and suction. Realistic and meaningful bleeding is a perennial problem for VR researchers. We have concluded that, visually, the actual drill representation needs only be quite simple, and it is felt that representing the spinning

(a) Typical Mastoidectomy surgical setup



(b) Mud formation



(c) Obscuring effects

Figure 1.1: Operation scene. On the left, the typical mastoidectomy surgical setup: the ENT surgeon looks at the region interested by the procedure via a stereoscopic microscope and holds in his hands a high speed burr and a sucker, that he uses, respectively, to cut the bone and to remove bone paste produced by the mixing of bone dust with water. On the right, there are typical examples of what is seen by the surgeon while performing mastoidectomy. In (a) it is clearly visible the paste created by the mixing of bone dust with water. If the paste and the water are not removed, they can obscure the field of view (b). Photos courtesy of Prof. Stefano Sellari Franceschini, ENT Surgery, Dept. of Neuroscience, University of Pisa.

of the cutter or diamond burr is unnecessary. What is considered necessary, from a functional standpoint, is an effective collision detection mechanism which not only copes with increased resolution as the virtual drill proceeds deeper into the temporal bone, but is also capable of generating error states when (for example) a large burr is inserted into a narrow drill site.

As for the nature of the technology required for displaying drill, drill site, bone, and so on, there is no conclusive evidence or support for the premise that the use of a stereoscopic system will aid performance in this case. Binocular viewing systems are deployed in the operating theatre and used by surgeons, and so binocular imaging should be available to the simulator. However, the wearing of any form of stereoscopic display, such as a head-mounted display or liquid crystal shutter glasses should be avoided. The surgeon or trainee does not want to use cumbersome eyewear that is not necessary for carrying on the real procedure. We make the hypothesis that, if the simulation achieves a reasonable level of fidelity, then the combination of high-resolution images and haptic feedback will, more than likely, suffice.

As well as the visual and 6-DOF input/3-DOF haptic feedback for drill simulation (including high frequency vibration), the training system might also be enhanced by the inclusion of audio effects. Some surgeons suggest that they are able to detect subtle changes in sound depending on the nature of the bone they are working with (eg. cortex *vs.* petrous). However, this quality is considered to be "overkill" in a training system such as that being considered here.

Chapter 2

Surgical simulation software kernel architecture

2.1 The decoupled simulation model

The results of the human factors analysis indicate that, to be able to feed the appropriate sensorial inputs to the human perceptual system, the system needs to produce data at two very different time-scales: about 15-20 Hz for the visual rendering, and around 1 KHz for the haptic response [AGG+02a]. The computations needed to obtain the haptic force response can be drastically simplified, since response forces can be computed by just considering a small neighborhood around the contact surfaces between surgical instruments and bones. The simulation of secondary effects and the visualization of the evolving operating theater requires, however, a larger computational effort.

We have exploited this difference in complexity and frequency requirements by modeling the simulator as a collection of loosely coupled concurrent components. Logically, the system is divided in a ”fast” subsystem, responsible for the high frequency tasks (surgical instrument tracking, force feedback computation, bone erosion), and a ”slow” one, essentially dedicated to the production of data for visual feedback (see figure 2.3). The “slow” subsystem is responsible for the global evolution of the water, bone dust and bone paste. These secondary effects can be considered purely visual, since they just contribute to visual clutter without producing important forces to be returned to the user. The algorithms used to control the simulations are local in character and they are structured so that they communicate only via changes in the relevant, local, substance densities. This arrangement leads naturally to a further break-up of the slow subsystem in components, each dedicated to the generation of a specific visual effect, and thus to a parallel implementation on a multiprocessor architecture. Figure 2.5 outlines the main components of the system, as implemented in our current prototype. The system runs on two multiprocessor machines connected with a 100 Mbit Ethernet link. The data is initially replicated on the two machines. The first machine is dedicated to the high-frequency tasks: haptic device handling and

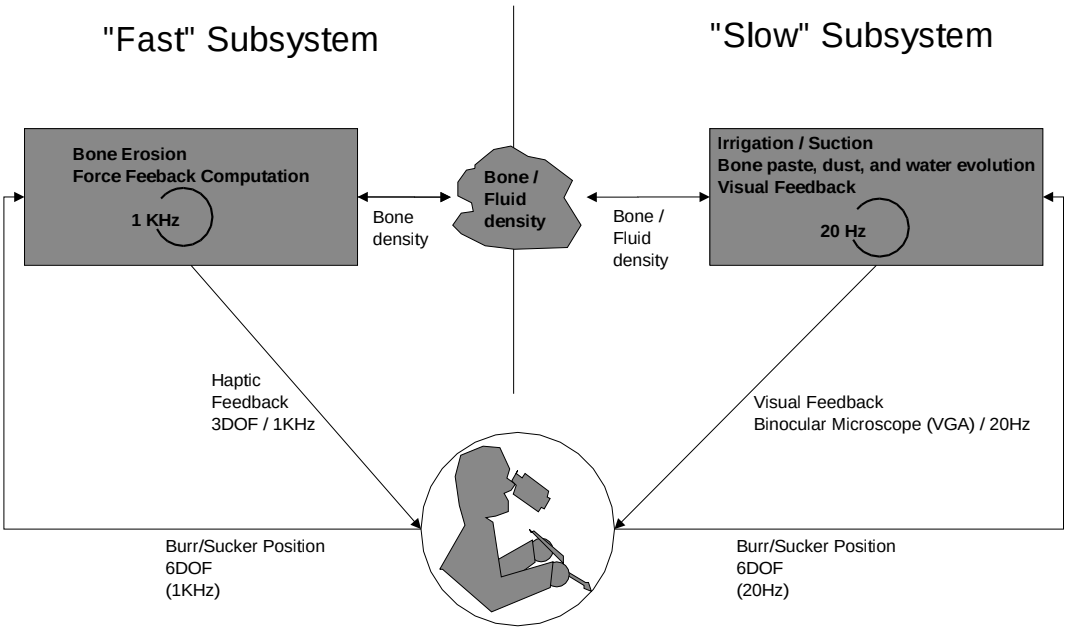


Figure 2.3: **Logical system decomposition.** The system is divided in a ”fast” subsystem, responsible for the high frequency tasks (surgical instrument tracking, force feedback computation, bone erosion), and a ”slow” subsystem, essentially dedicated to the production of data for visual feedback.

bone removal simulation, which run at 1 KHz. The second machine concurrently runs, at about 15-20 Hz, the low-frequency tasks: bone removal, fluid evolution and visual feedback. Since the low-frequency tasks do not influence high-frequency ones, the two machines are synchronized using one-way message passing, with a dead reckoning protocol to reduce communication bandwidth.

A major design decision is the definition of the actual representation of the data. We have chosen to consistently use a voxel-based volumetric approach, where the model is represented by a regular array of material labels with associated density. This representation has a number of advantages: first, since data organization is the same as the one of the acquired data, errors introduced by reformatting and/or surface extraction are avoided; second, local editing and point location operations can be implemented at low cost; finally, an array-based data structure can be shared very efficiently between concurrent processes. This representation, however, brings important challenges: the number of contacts between voxel-based volumetric objects poses a problem for calculating collisions response [GSMF97]; fluid-dynamic computations scale with the cube of volume dimensions; rendering a dynamic volume under real-time constraints is an inherently complex task, since a large number of volume elements may contribute to the final image.

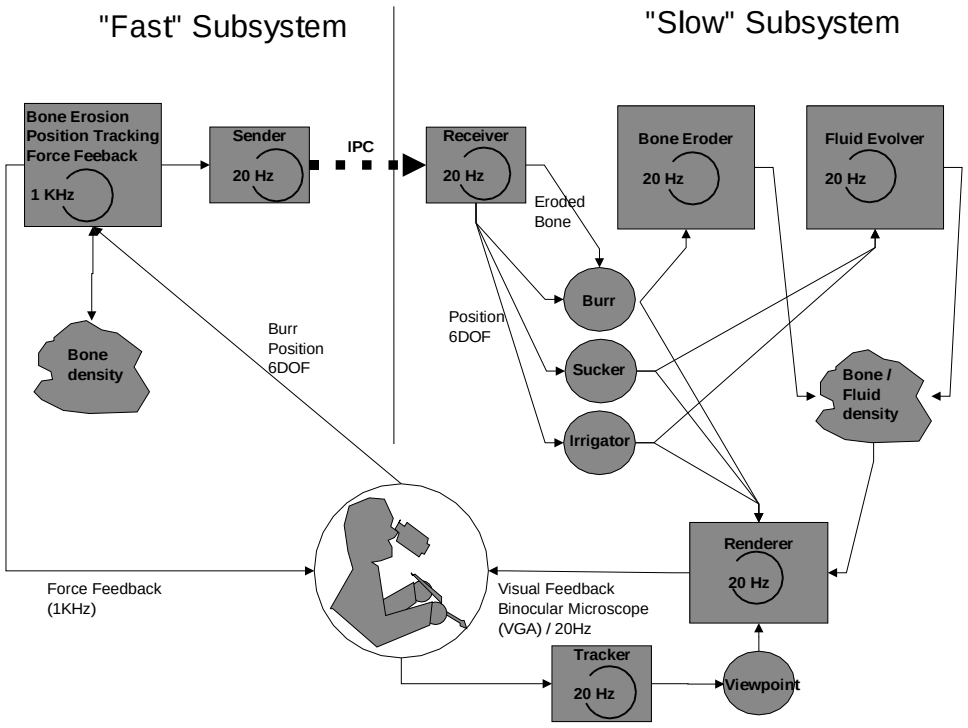


Figure 2.5: **Decoupled simulation architecture.** The system uses a volumetric approach, with the initial configuration of the model directly derived from patient CT data. The data is initially replicated on the two machines. The first machine is dedicated to the high-frequency tasks: haptic device handling and bone removal simulation. The second machine concurrently runs at 10-20 Hz the low-frequency tasks: bone removal, fluid evolution and visual feedback. The two machines are synchronized using one-way message passing with a dead reckoning protocol.

The technical solutions implemented in our prototype are presented in the following section.

2.2 Burr-bone interaction and haptic feedback

A detailed mechanical description of the cutting of material by a rotating burr is complicated because it involves: the tracking of the continuously changing free surface of the material being cut; the impact of the burr blades on the surface; the resulting stress distribution in the material; and the consequent plastic deformation and break–up. In the general engineering context these problems are solved by using experimentally determined curves, but, for the specific case of bone burring, there are no publicly available data. Furthermore, in the specific context of haptic feedback, one cannot apply the standard methods found in the mechanical engineering literature for the simulation

of milling. In fact, an haptic feedback system is driven by an open–loop controller that needs to rapidly evaluate a reasonable response force for arbitrary tool penetrations.

To circumvent these complications, we have developed a simplified model, originally described in [AGG+02d], based on a limited number of parameters that are, at the moment, tuned by trial and error following the opinion of expert surgeons as feedback. In this deliverable, we also present adaptive techniques that trade simulation quality with speed in order to meet real-time constraints.

The basic assumption underlying our model is that the burr bit is moving relatively slowly with respect to the time scale of the haptic feedback loop and that one can estimate the elastic forces exerted by the bone by geometrically characterizing the region of bone intersected by an idealized sphere representing the burr tip.

Specifically, we model the burr bit, B, with a sphere of radius R centered at $\mathbf{R}_b$, and consider the first two moments of the bone mass density, $\rho(\mathbf{r})$, contained in B.

$$m_0 = \int_{r<R} dr^3 \rho(\mathbf{r}), \mathbf{m}_1 = \int_{r<R} dr^3 \rho(\mathbf{r})\mathbf{r}. \tag{2.1}$$

The direction of the local normal, $\hat{\mathbf{n}}$, to the bone surface can then be estimated as $\hat{\mathbf{n}} = -\mathbf{m}_1/|m_1|$, and from the amount of mass contained in B, m_0, we can derive an effective "penetration depth" h as the smallest positive solution of

$$m_0 = \pi\rho_0 R^3 (\frac{h}{R})^2 (1 - \frac{h}{3R}) \tag{2.2}$$

where ρ_0 is the "solid" bone reference density.

We can now write an expression for an effective force $\mathbf{F}_e$, that is supposed to model the elastic response of the bone to the impinging burr.

$$\mathbf{F}_e = c_e R^2 (h/R)^{3/2} \hat{\mathbf{n}}, \tag{2.3}$$

where c_e is a dimensional constant, that, as far as this model is concerned, describes the elastic properties of the material. In the limit of $h/R << 1$, eq. (2.3) is consistent with Hertz's contact theory [LL86].

Typical burr radii are between 1 mm and 5 mm, while the typical speed at which the burr bit is moved is < 100 mm/s [AGG+02b]. Given that the haptic device acquisition period is 1 ms, the burr bit will typically move a distance of the order of a few percents of its radius. Therefore, it is reasonable to compute interaction forces by checking collisions after the fact, rather than trying to predict them in advance.

For a given response force F_e^* one can invert eq. (2.3) to obtain, assuming that $h/R << 1$,

$$m_0^* \approx \pi\rho_0 R^{1/3} (F_e^*/c_e)^{4/3}. \tag{2.4}$$

Hence, the amount of computational work needed to resolve, say, a zero force threshold increases only slowly with the burr radius. On the other hand, in typical burr usage one applies a force on the burr so that it will have an instantaneous erosion surface that

scales as R^2. Since the contact surface, S, of the burr with the bone, again for small h, is proportional to hR, this corresponds to a mode of operation where the force applied by the user on the burr is adjusted to maintain h roughly proportional to R, $h \approx \alpha R$. With this assumption,

$$\frac{dF_e}{d\, m_0} = \frac{3}{4}\frac{c_e}{\pi\rho_0}\frac{1}{R}(\frac{h}{R})^{-1/2}\frac{1}{1-\frac{h}{2R}}, \tag{2.5}$$

and

$$F_e = c_e R^2 \alpha^{3/2}, \tag{2.6}$$

therefore

$$\frac{1}{F_e}\frac{dF_e}{d\, m_0} \approx \frac{3}{4}\frac{1}{\pi\rho_0}\frac{1}{R^3\alpha^2}. \tag{2.7}$$

For a given accepted error ratio β in the haptic force, $\beta = \Delta F_e/F_e$, we can estimate the accepted error for m_0, Δm_0, to be

$$\Delta m_0 \approx \beta\alpha^2 R^3. \tag{2.8}$$

Therefore, in this mode of operation, we can maintain the relative error in force estimation constant at a small computational cost even for increasing burr radius R. In fact, we are allowed to increase linearly with R the discretization scale, ℓ, used in computing the integrals in eq 2.1.

In the following, we will describe a computational method that exploits these observations to compute $\mathbf{F}_e$ with a computational cost that grows slowly with R and is well within the time constraints, 1 msec total for force estimate and bone erosion, imposed by the haptic feedback device.

The new method completely overcomes the limitations to small burr sizes of the technique used in [AGG+02d].

2.2.1 Multi-scale spatial description

The integrals requested by eq. 2.1 can be easily computed using a multi–resolution volumetric description of the region of interest.

We partition the volume of interest using an octree, with the leaves of the octree that directly refer to the scene voxels, and the coarsest level to the whole scene. In an initialization phase, starting from the leaves, we precompute, for each octree block, I, the local values of m_0^I and $\mathbf{m}_1^I$. The zeroth moment of the mass contained in block I is simply the sum of its values at the block children $\{I, k\}$, $m_0^{\{I,k\}}$. To compute $\vec{m}_1^I$ we use the center of mass decomposition rule

$$\mathbf{m}_1^I = \sum_k [\mathbf{r}_I^k m_0^{\{I,k\}} + \mathbf{m}_1^{\{I,k\}}], \tag{2.9}$$

where $\mathbf{r}_I^k$ is the vector that goes from the center of block I to the center of its child k.

The algorithm used to estimate m_0 and $\mathbf{m}_1$ is then the following. At each haptic cycle, we descend the octree until we find blocks that are either fully contained or partially intersecting the burr sphere. If they are fully contained, we add their contribution to m_0 and $\mathbf{m}_1$; if they are partially intersection, we compare the block size with ℓ and if it is larger we refine; otherwise, we add the partial volume contributions

$$\Delta m_0 = \frac{\Delta V}{V_I} m_0^I \tag{2.10}$$

$$\Delta \mathbf{m}_1 = \frac{\Delta V}{V^I}(\mathbf{R}_b - \mathbf{R}_c)(m_0^I + \mathbf{m}_1^I \cdot (\mathbf{R}_c - \mathbf{R}^I) + O((\ell/R)^2), \tag{2.11}$$

where ΔV is the volume of the region of intersection between block I and the sphere, $\mathbf{R}_c$ is the position of the center of mass of the latter intersection, and $\mathbf{R}^I$ is the position of the center of block I. In the current implementation of the algorithm, both ΔV and $\mathbf{R}_c$ are approximated by replacing the block with a sphere of equal volume (see [AGG+02d] for details).

Therefore, at the cost of a minor computational overhead in the precomputing and update (see below the discussion on erosion), we are able to estimate m_0 and $\mathbf{m}_1$ with a computational cost that grows as most as $h^2 R/\ell^3$. Moreover, the availability of the precomputed $\mathbf{m}_1^I$ moments allow us to estimate the contribution of partially overlapping blocks at a higher order to what would have been possible using only m_0^I. We are thus allowed to use larger values for ℓ.

2.2.2 Multi-scale erosion

Erosion, i.e. material removal in response to burring, is modeled as a position dependent erosion rate described by f, an erosion shape function,

$$\frac{d\rho(\mathbf{r})}{dt} = \alpha f(r/R)\rho(\mathbf{r}); \tag{2.12}$$

where, again, $\mathbf{r}$ is measured from the center of B, and α is an appropriate dimensional constant. f is constrained to have a maximum at $r/R = 0$ and to be null for $r/R > 1$. In a previous work,[AGG+02d], erosion was modeled by assuming that all the power spent by working against the frictional forces on a "contact surface" element of the bone would have gone toward the erosion of the bone material on the surface. The resulting expression for the local mass derivative was, however, rather complex and computationally expensive. Eq. (2.12) provides essentially comparable results at a much lower computational cost.

From the point of view of the implementation, in our model the bone is described as a collection of voxels, each one containing up to 255 values of bone occupation. To accommodate for a wide range of erosion rates using only 8 bits, we convert the rate of erosion given in Eq. (2.12) to a probability that the value of the voxel at position $\mathbf{r}$

will be reduced by one at next time step. A Russian roulette scheme is then used for deciding whether to fully erode a bit (i.e. remove 1/255th of the mass of a full voxel) or not.

To find the voxels that should be eroded, we integrate the following modifications to the octree descent algorithm introduced above. When we identify a block as contained in the burr, we descend down to all the leaves and erode the voxels using the probabilistic version of Eq. (2.12). When the block is instead only partially contained in the burr, we continue recursion until we find completely contained sub blocks and then proceed as above. If we reach a leaf which is only partially contained, the erosion probability is scaled by the overlap fraction before testing for erosion. In descending the octree we keep track of the number of voxels touched while visiting a node children. If it changes, we perform an update of the node value from its children values using the same scheme used for octree construction (i.e. pulling moment updates from octree leaves up to the root).

2.2.3 Other contributions to the haptic response

Together with the elastic force $\mathbf{F}_e$ defined in eq. (2.3), we also compute a frictional force, $\mathbf{F}_\mu$, that is supposed to model the friction forces that oppose burr rotation when the latter is in contact with the bone material; and an impact force, $\mathbf{F}_i$ that can be thought as what would be the response of the bone material if it were modeled as a collection of unconnected point masses swept by the moving burr sphere.

$$\mathbf{F}_\mu = c_\mu R^2 (h/R)(\mathbf{m}_1/m_0) \times \omega \tag{2.13}$$

$$\mathbf{F}_i = -(c_i R^2)(h/R)\mathbf{V} \tag{2.14}$$

where we have introduced ω, representing the burr angular velocity vector, and $\mathbf{V}$ the velocity of the burr center.

2.3 Secondary visual effects

Although the presence of the water/paste mixture is essentially irrelevant with respect to the interaction between the burr and the bone, its presence cannot be neglected in the creation of the visual feed–back, because its "obscuring" effects constitute the principal cue to the user for the use of the suction device[AGG$^+$02b].

A direct, "physically correct", simulation of the dust-water system would require, to be able to capture all the dynamically relevant length scales, a very fine spatial resolution and it would be computationally incompatible with the real–time requirements of the simulation. For this reasons, secondary effects were mostly neglected in prior bone burring simulations. The Ohio Virtual Temporal Bone Dissection simulator simply removes voxels by making them transparent [WBS$^+$00, BSWS01]. Localized bleeding is simulated by coloring in red the voxels close to the burr bit. Our system [AGG$^+$02d] exploits the difference in frequency requirements of the visual and

haptic simulations by running a rule-based particle system simulator in parallel with the bone dissection simulator. The method is able to provide a crude visual approximation of bone debris accumulation, blooding, irrigation, and suction. In this deliverable, we report improvemements on the technique presented earlier, obtained by introducing a time-critical particle evolution method that trades simulation quality with time.

For the computational reasons, we are modeling the dust/fluid dynamics using what essentially amounts to an hybrid particles-volumetric model, inspired by previous work on particle systems and sandpiles [RS99, LM93]. In this scheme, particles are created by the irrigator, which injects water particles, by blood spots and vessels, that inject blood particles, and by the burr during erosion, that converts bone to bone dust particles. All particles move ballistically when in empty space, and interact with the other materials according to a set of rules that ensure that only a single particle may occupy a given voxel at given time (see [AGG+02c]). Particles are deleted when they exit from the operation site or when they are sucked by the suction device.

The computational cost of update in this scheme is essentially constant per particle and, thus, the total computation cost would naively grow linearly in the number of particles and quickly degrade the real-time performance of the system. To avoid this problem, we are using a time-critical evolution algorithm designed to trade simulation quality with speed. The idea behind the algorithm is to concentrate resources on the visually most important parts of the simulation, by controlling both individual particles update rates and total number of particles.

The update rate control methods associates to each particle an update rate proportional to the particle speed. To avoid the costs associated to sorting the particles, the particles are divided in groups, $\{G_i\}$, so that all the particles in group G_i have speed v, measured in units of a predefined reference maximal velocity scale, in the range $2^{-i} \leq v < 2^{-(i+1)}$. Particle velocities are clamped so that they cannot be larger that the maximal velocity scale. At each evolution time step we randomly select $\{a_i\}$ particles from each group and, for each selected particle, integrate the motion from its last recorded time of update to the current time.

The effective time step for particles in group G_i is then $(dt)_i = n_i/a_i(dt)_\mu$ where n_i is the number of particles in group G_i and $(dt)_\mu$ is the actual simulation time step. The selection counters $\{a_i\}$ are chosen so that, on average, particles in channel i will move with a time step $(dt)_i = 2(dt)_{i-1}$, and thus

$$\frac{a_{i+1}}{n_{i+1}} = \frac{1}{2}\frac{a_i}{n_i}. \tag{2.15}$$

The total computational cost for one time step will then be $W = w\sum_i a_i$ where w is the average cost per particle update, which is measured at run time by the simulator. Using the equation above we find that, when all the $n_i > 0$,

$$A = \sum_i a_i = \frac{a_0}{n_0}\sum_i \frac{n_i}{2^i}. \tag{2.16}$$

Therefore, for given W, w, and n_i, we can reconstruct the required a_i. The case $n_j = 0$ for some j is a trivial generalization of the above.

Given a reasonable approximation of w, the update rate control algorithm is guaranteed to meet timing constraints and to probabilistically move the particles with the largest visual error. If the update rate of the particle system falls below a specified threshold (currently, if we move less that 10% of the particles per step), we reduce the particle count by removing the "less important" ones. The importance of a particle is currently inversely proportional to the distance from the current lookat point of the microscope and to the particle velocity.

2.3.1 Sample–Estimate-Hold Interface

A direct transmission of the computed forces to the haptic device is, in the case of "almost rigid" contacts, usually plagued by mechanical instabilities. The typical solution for this problem is the introduction of an artificial, "virtual", coupling between the haptic device and the virtual environment [Col94, AH99].

In our system, we use a *sample–estimate–hold* approach [ESJ97] to remove the excess energy injected by the standard zero–order hold of force employed by the haptic device drivers. With this technique, we compute the force that is sent to the haptic device based on the previous zero–order representations produced at regular intervals by our burr–bone interaction model. This new value of force, when held over the corresponding sampling interval, approximates the force–time integral more closely than the usual zero–order hold [ESJ97].

2.4 Real–time visual rendering

The surgical simulator must achieve the visual illusion of animation and responsiveness by rapid successive presentation of a sequence of static images of the evolving operating theater as seen from the surgical microscope. Since humans are very sensitive to synchronization problems between synthesized and real-world sensory input, it is of primary importance for the visual rendering subsystem to operate within the timing constraints imposed by the human perceptual system (i.e. latency of less than 300 ms, and frequency above 10-15 Hz [MZ92, HD91, YJN$^+$95]).

We reach this goal using a parallel processing approach, which exploits the capabilities of current graphics PC architectures. In our system, the renderer is totally decoupled from the simulator and the tracking system, and runs at his own frequency. At each rendered frame, the following actions are taken:

1. the time of presentation of the frame is predicted;

2. the Z and color buffer are cleared;

3. the position/orientation of the surgical microscope at the end of the frame is extrapolated from the latest sensor data; the camera view/projection matrices are set accordingly;

4. the position/orientation of the surgical instruments at the end of the frame are extrapolated from the latest sensor data; a polygonal representation of the surgical instruments is rendered to the Z and color buffer;

5. the simulation state is presented by projecting and compositing onto the image the elements of the volumetric data representation, which is shared with the simulator;

6. the image is presented;

This technique relies on the ability to rapidly render a good quality view of a continuously changing scalar volume. Our algorithm, based on texture mapping and back-to-front composition of volume slices, maximizes parallel efficiency by asynchronously performing volume rendering while the simulator is updating the volume.

2.4.1 Shaded direct volume rendering of dynamic volumes

In direct volume rendering, images are produced by integrating along selected projectors the value of a continuous emission/reflection/absorption volume function reconstructed from discrete sampling points [Max95]. By manipulating the mapping from values of the original volume data to emission, reflection, and absorption coefficients, various effects can be achieved, including isosurfaces and opaque objects. In our case, the volume is a regular 3D grid containing at each voxel a material identifier (e.g. air, bone, dust, water, blood). The latter is continuously reassigned by the simulation, that is running in parallel to the rendering process. Rendering such a dynamic volume under real-time constraints is particularly challenging.

A number of authors have proposed to exploit texture mapping and rasterization hardware to render scalar volumes at interactive speeds [CN94, CCF94, GL94, VK96, Kul96]. These techniques are based on uploading the scalar volume to texture memory prior to rendering object-aligned or view-direction-aligned textured volume slices. One of the major limitations of these methods is their inability to efficiently implement surface illumination models, since texture lookup is based only on data values and not on gradient information. Various authors have proposed alternative techniques for supporting hardware-accelerated direct volume rendering with shading [VK96, WE98, RSEB$^+$00, EKE01]. However, this comes at the expense of performance and texture memory overheads, since the proposed techniques require multiple passes through the rasterization hardware and/or precomputation of gradient volumes. This is unacceptable in our case, since the volume is continuously varying, and thus we cannot compute and reload gradient maps.

In our approach, a fast approximation of the shading equation is computed on the fly by the graphics pipe-line directly from the scalar data. We do this by exploiting the possibilities offered by multi-texturing with the register combiner OpenGL extension, that provides a configurable mean to determine per-pixel fragment coloring [Kil00]. The extension is available on commodity graphics boards (e.g., NVIDIA GeForce series).

To simulate shading effects from contour surfaces at sharp changes in a scalar volume function, a common approach [Max95] is to use the opacity gradient to measure surface "strength", and to shade the volume using a simple Lambert diffuse shading formula multiplied by the strength, giving, for a single directional light:

$$I(x,y,z) = (c_a + c_d \left| \vec{\nabla} k_m^\alpha(x,y,z) \cdot \vec{l} \right|) \cdot k_m(x,y,z) \tag{2.17}$$

where c_a and c_d are the ambient and diffuse RGBA intensities of the light, k_m is the material RGBA color, and $\vec{l}$ is the direction of the light. If we assume that the light direction is coincident with the volume coordinate axis which is pointing towards the viewer (e.g., the local Z axis), we need to compute only a single component of the gradient (in the example, the Z component). This approximation is acceptable in our case, because of the particular microscope setup which limits the viewer to almost frontal views [JTP+01, AGG+02a]. The shading formula becomes, using a forward difference approximation of the gradient:

$$\frac{\Delta k_m^\alpha}{\Delta z} = \left| \frac{k_m^\alpha(x,y,z+\Delta z) - k_m^\alpha(x,y,z)}{\Delta z} \right| \tag{2.18}$$

$$I(x,y,z) = (c_a + c_d \frac{\Delta k_m^\alpha}{\Delta z}) \cdot k_m(x,y,z) \tag{2.19}$$

This equation can be implemented in the graphics hardware by programming the register combiners (see figure 2.6), leading to an efficient shaded volume rendering algorithm in which all computation is performed by the graphics hardware starting simply from the scalar volume.

At the beginning of the procedure, the material table, which maps the material identifiers in the volume to the RGBA colors k_m, is loaded in the shared texture palette. The register combiners are then configured as in figure 2.6 to implement slice interpolation and fragment shading. The volume is then traversed back-to-front, and the 2D slices are sequentially loaded into texture memory, alternating between texture 0 and texture 1. For each pair of slices, a number of intermediate slices are synthesized by rendering planar polygons and storing the interpolation factor in one of the constant color registers. For each fragment, general combiner 0 generates the color of the intermediate slice by interpolating between the front and back slice using the given interpolation factor, general combiner 2 computes the opacity gradient, and the final combiner computes the fragment's final RGBA color as in equation 2.18.

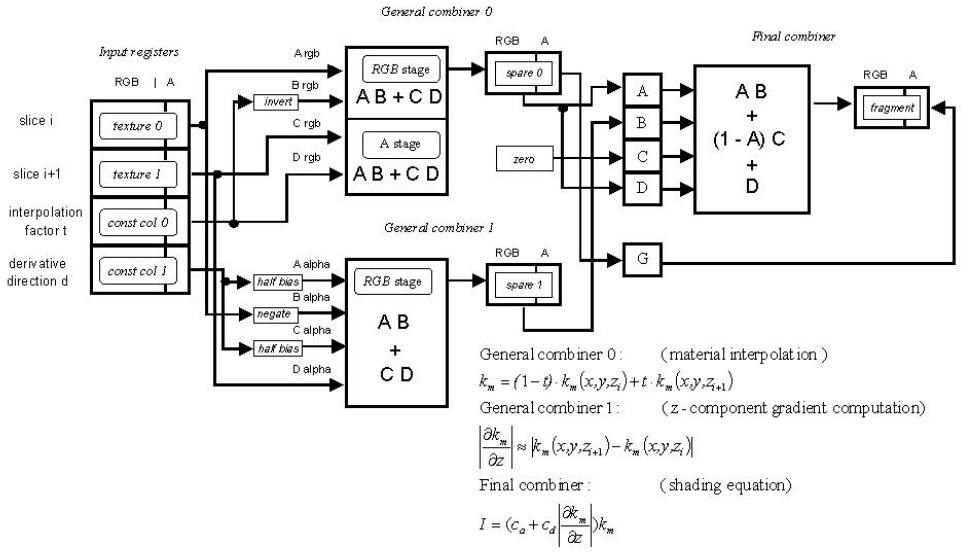


Figure 2.6: **OpenGL combiner setup.** The final combiner blends the interpolated slice value from the first combiner with the shading value form the second combiner.

This procedure is extremely efficient, since all the computation in performed in parallel in the graphics hardware and no particular synchronization is needed between the renderer and the process that is modifying the dataset. Only a single sweep through the volume is needed, and volume slices are sequentially loaded into texture memory on current standard PC graphics platform using AGP 4X transfers, which provide a peak bandwidth of 1054 MB/s. A 256x256x256 dynamic volume using 8 bit material identifiers may thus potentially be transfered to texture memory at over 60 fps. Only two slices need to be present in texture memory at the same time.

2.4.2 Reducing fill-rate bottleneck

Pixel fill-rate is the major limiting factor when using a texturing approach to volume rendering. In zoom rendering, an appropriately down-scaled image is rendered in the back buffer and then enlarged and copied to the front buffer [MSG95]. This way, delays associated with buffer swap synchronization are avoided, and the number of pixels filled during volume rendering is reduced. In our implementation, the copy and zoom operations are implemented by copying the reduced size image in texture memory and then rendering a textured polygon in the front buffer. This way, sophisticated texture interpolation algorithms can be used to reduce the artifacts caused by magnification.

Zoom rendering is particularly useful in our application, because the pixel resolution is much larger than the resolution of the data that is displayed in the window.

Chapter 3

Hardware System Configuration

A prototype system, based on the techniques discussed above, is running on a dual PC platform. Our current configuration is the following:

- a single-processor PIV/1400 MHz with 256 MB PC133 RAM for the high-frequency tasks; two threads run in parallel: one for the haptic loop (1KHz), and one for sending volume and instrument position updates to the other machine;
- a dual-processor PIII/800 MHz with 512 MB PC800 RAM and a NVIDIA GeForce 4 Ti4600 running a Linux 2.4 kernel for the low frequency tasks; three threads are continuously running on this machine: one to receive volume and position updates, one to simulate bone removal and fluid evolution, and one for visual rendering;
- a Phantom Desktop haptic device for the dominant hand; the device is connected to the single processor PC. It provides 6DOF tracking and 3DOF force feedback for the burr/irrigator;
- a Phantom 1.0 haptic device for the non-dominant hand; the device is connected to the single processor PC. It provides 6DOF tracking and 3DOF force feedback for the sucker;
- an n-vision VB30 binocular display for presenting images to the user; the binoculars are connected to the S-VGA output of the dual processor PC.

Figure 3.1: The current Ierapsi surgical simulator set-up. Note the Phantom Desktop haptic device for the dominant hand; the Phantom 1.0 haptic device for the non-dominant hand and the n-vision VB30 binocular display. The system is driven by, not shown in figure, a single-processor PIV/1400 MHz with 256 MB PC133 RAM (haptic loop) and a dual-processor PIII/800 MHz with 512 MB PC800 RAM and a NVIDIA GeForce 4 Ti 4600.

Chapter 4

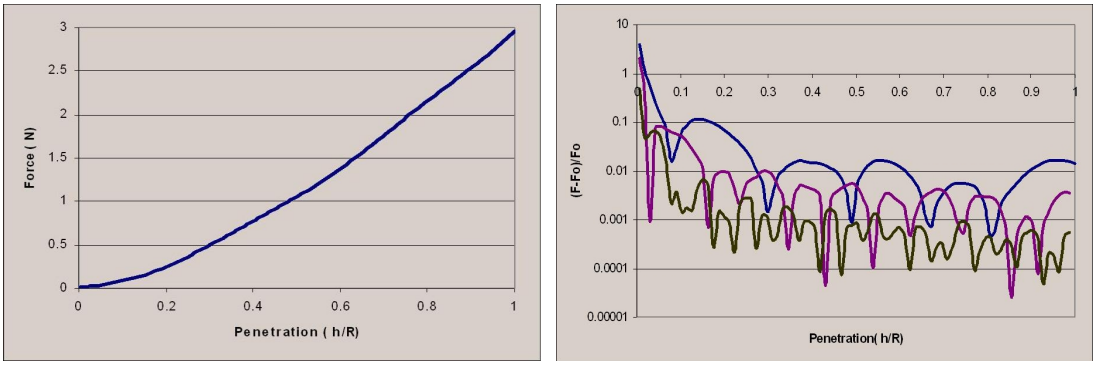
System Evaluation

4.1 General system performance

The performance of the prototype is sufficient to meet timing constraints for display and force-feedback, even though the computational and visualization platform is made only of affordable and widely accessible components. The volumetric datasets used to represent the region where the operation takes place incorporate information on bone and on the noble structures that should be avoided while performing the simulated operation. To obtain these datasets, we need to combine information from several modalities that contain complementary data, specifically, computed tomography (CT) provides high spatial resolution bone images whilst magnetic resonance imaging (MRI) provides images of soft tissues. In the simulations reported in this report we are using a volume of 256x128x128 cubical voxels (0.3 mm side) obtained by manually adding soft-tissue information to an high resolution CT scan. The resolution of the volume is the same as the original CT data. The force-feedback loop is running at 1 KHz using our multiresolution force evaluation model for force computations. Shaded volume rendering of dynamic volumes currently takes 70 ms per frame (i.e. over 14 frames per second) using 256 depth slices on an 800x600 window with 16 bit color and 2X zoom rendering.

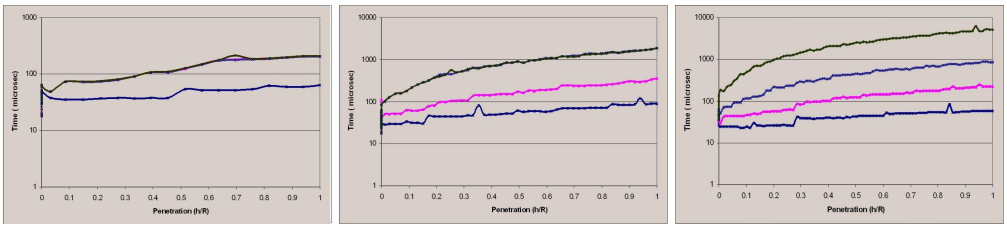
4.1.1 Multi-resolution Force Evaluation

Figure 4.1(a) shows the reaction of the virtual bone against burr penetration, using different burr-sizes and different accuracy parameters. The computations are done in absence of erosion, $\alpha = 0$ in equation 2.12, and using the actual force evaluation kernel of the force–feedback loop with a volume composed of cubical voxels with 0.3 mm side. The figure shows the "elastic" response of the material when using two different burr sizes ($R = 1.0mm, R = 5.0mm$), which correspond to a standard polishing burr tip and a large initial burring tip. The force has been computed using the mono-resolution algorithm, as well as three different accuracy settings of the multi-



(a) Elastic force (b) Relative error

Figure 4.1: Virtual bone reaction against burr penetration and relative error in force evaluation introduced by the multi-scale algorithm.



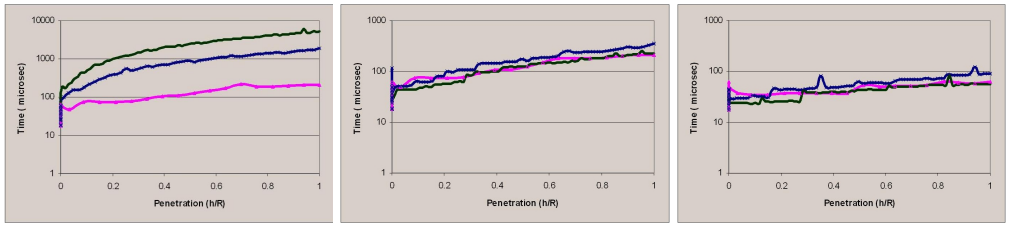
(a) $R = 1$mm (b) $R = 3$mm (c) $R = 5$mm

Figure 4.2: Time required to compute the forces of figure 4.1(a) compared by radius.

resolution algorithm, corresponding to ($\ell = 0.1R, \ell = 0.3R, \ell = 0.5R$). The graphs clearly show that the mono-resolution and the multi-resolution version of the algorithm are in agreement. In figure 4.1(b), we report how the the relative error with respect to the reference mono-resolution solution changes with penetration. As it can be seen from the figure, it is typically of the order of few percents or below. The oscillations in the curves are due to resonances between the burr position and the octree grid used to compute the forces.

In figure 4.2, we report the wall clock time required by the force computation kernel to compute the forces of figure 4.1(a). Each subfigure shows the wall clock time required for different values of the burr radius and for different resolution scales. For $R = 1$mm, as expected, there is no appreciable between the mono-resolution results and the multi-resolution ones for $\ell = 0.1R, \ell = 0.3R$, while the $\ell = 0.5R$ is faster. For $R = 3$mm $\ell = 0.1R$ is still of the same order of the voxel size, 0.25mm, while the $\ell = 0.3R$ and $\ell = 0.5R$ are now clearly faster than the monoresolution case.

In figure 4.3, we show the growth of the computational cost for a given resolution



(a) $\ell = 0$ (b) $\ell = 0.3R$ (c) $\ell = 0.5R$

Figure 4.3: Time required to compute the forces of figure 4.1(a) compared by resolution scale.

scale and different radius values, $R = 1, 3, 5$mm. The figures show the growth of the computational cost for a given resolution scale and different radius values, $R = 1, 3, 5$mm. It is clear from the figures that the mono-resolution algorithm is limited to $R < 2$mm, the computational cost of the multi-resolution algorithm grows very slowly with R. and always easily meets the 1 ms haptic feedback time constraint.

4.1.2 Visual feedback

Figure 4.4 shows a few frames taken from the live recording of a typical virtual bone dissection sequence performed in the mastoid region. The tool on the left is the suction device. It interacts with the scene by simply removing all the particles within a certain radius from its tip. The tool on the right is the burr, connected with the irrigator. All the visual feedback in the images is provided by the particle simulator, that transforms removed bone into bone dust particles, injects water and blood particles, and removes the resulting paste. The selected images correspond to what is seen by the user in the microscope. On our PIII/800 simulator machine, with standard simulation settings, we use ten channels for velocity sorting and impose a simulator time step of $(dt)_\mu = 10$ ms and a visual feedback rate of ten to twenty frames per second. The total number of particles in the scene is in the order of the tens of thousands, with a per-particle update time w of a few microseconds.

4.2 Test sessions

We have extensively tested the virtual surgical training system in collaboration with surgeons of the Department of Neuroscience of the University of Pisa. In particular, contact model parameters and erosion factors have been tuned according to their indications and there is consensus that they represent a good approximation of reality. Using the tuned system, surgeons can perform complete virtual surgery procedures with satisfactory realism.

The possibility of recording dynamic values of a surgical training session provides new opportunities for the analysis and the evaluation of procedures. Different surgical procedure could be recognized by the system and it becomes possible to use the recorded values also to compare the behavior of expert surgeons and trainees in order to evaluate surgical skills.

Current available data show consistency between different training sessions of the same user. Average forces exerted by burr are between 0.7 and 1.3 N for the expert surgeon and between 0.8 and 1.1 N for trainees, while average tool velocities are between 8.0 and $12.0\ m/sec$ for the expert surgeon and 10.0 and $17.0\ m/sec$ for trainees.

In order to evaluate the possibility of characterizing different procedures according to dynamical parameters computed by the simulator, we recorded all the parameters (i.e. burr and sucker positions and velocities, force vectors, voxels removed) during a series of simulated mastoidectomy procedures.We analyzed four steps of the mastoidectomy procedure. In the first, the surgeon removes the cortex. The drill is applied to the mastoid cortex immediately posterior to the spine of Henle and draws two perpendicular cuts, the first along the temporal line and the second toward the mastoid tip. Then the mastoid cortex is then removed in a systematic fashion of saucerization.
Figure4.5 shows a snapshot of the scene viewed by the trainee during this step and on the right plots of the force module and of the material removed as a function of time. The second step is the cavity saucerization: before a deeper penetration in the antrum, it is necessary to perform a wide cortical removal and the posterior canal should be thinned so that the shadow of an instrument can be seen through the bone when the canal skin is elevated. Snapshot and plots relative to this step are shown in Figure4.6. In the next phase considered there is the identification of the mastoid antrum. It can be identified as a larger air-containing space at whose bottom lies the basic landmark of the smoothly contoured, hard, labyrinthine bone of the horizontal semicircular canal. The localization of this canal allows exposure of the fossa incudis, the epitymphanum anteriorly and superiorly and the external genu of the facial nerve medially and inferiorly. Snapshot and plots relative to this step are shown in Figure4.7. The final part of the basic mastoidectomy is represented in Figure4.8. During this step several landmarks are identified, and also the facial recess area is discovered. Force and voxel removal plots show that each step in the surgical procedure can be characterized by different actions. In the first step, the force plot presents evident peaks and valleys due to the necessity of creating holes to start the bone removal. In the second step the force is more continuous and not too high. During the mastoid antrum exposure the force is irregular and reaches higher values, up to 3N. The removal rate is similar, about 10.000 voxel removed per second. Finally the last considered phase is characterized by large pauses where there is no voxel removal and even when removal is present its rate is lower than in the previous steps, indicating that critical sites have been reached and consequently burring movements are more careful and accurate.

These facts can be pointed out just taking statistical values relative to the considered steps displayed in figures 4.9. It is possible, for example, to distinguish two phases with high average values of force and bone removal and two with lower values.

The two phases with high bone removal can be distinguished by the average burr velocity: in the mastoid cortex removal, where the user try to start new paths for the bone removal, the velocity is limited, while in the mastoid atrium exposure, where the user removes small quantities of material burr's movements are much faster. The cavity saucerization and the facial nerve identification phases, characterized by lower force values can also be distinguished by correlating with the burr bit movements speed. In fact, in the first phase the burr moves quickly along already determined paths, while in the second it is moved slowly – and carefully – since there is an high risk of damaging the facial nerve.

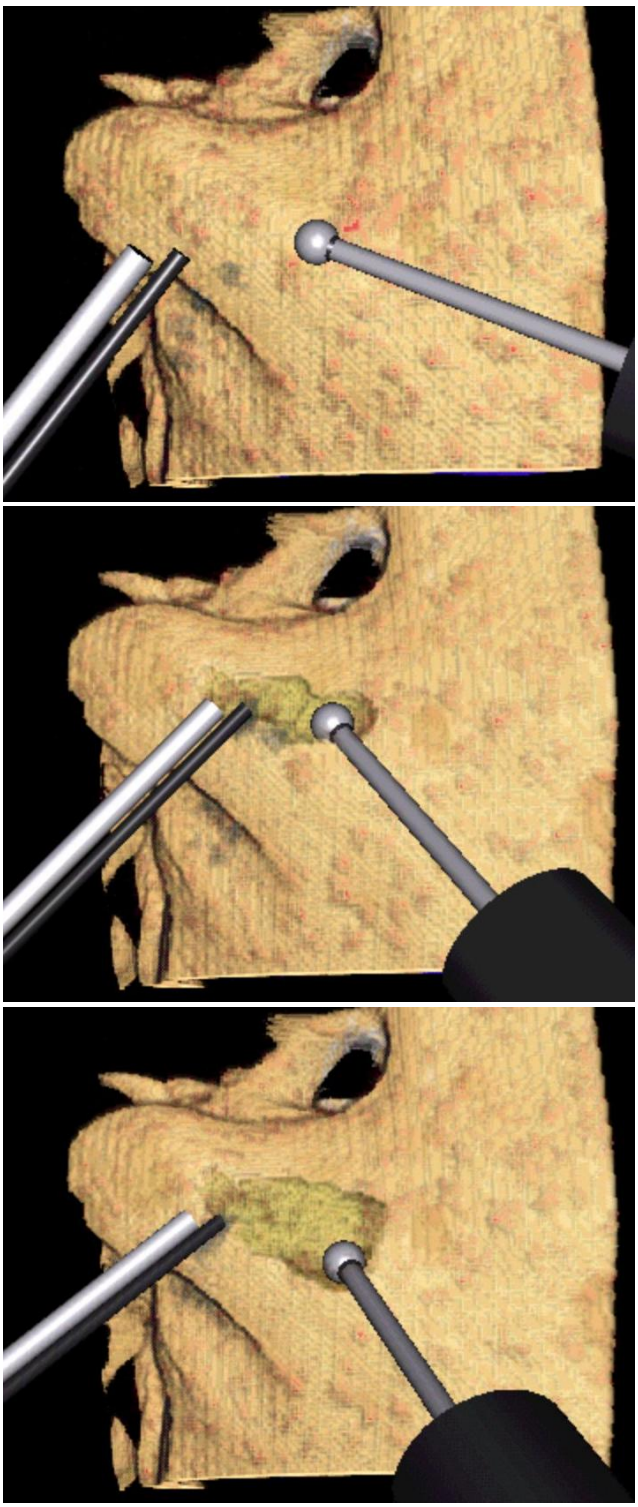


Figure 4.4: A virtual burring sequence performed in the mastoid region. Using our adaptive techniques, haptics simulation and bone removal run at 1KHz, while visual simulation, that includes blooding, debris accumulation, and suction, runs at 20Hz.

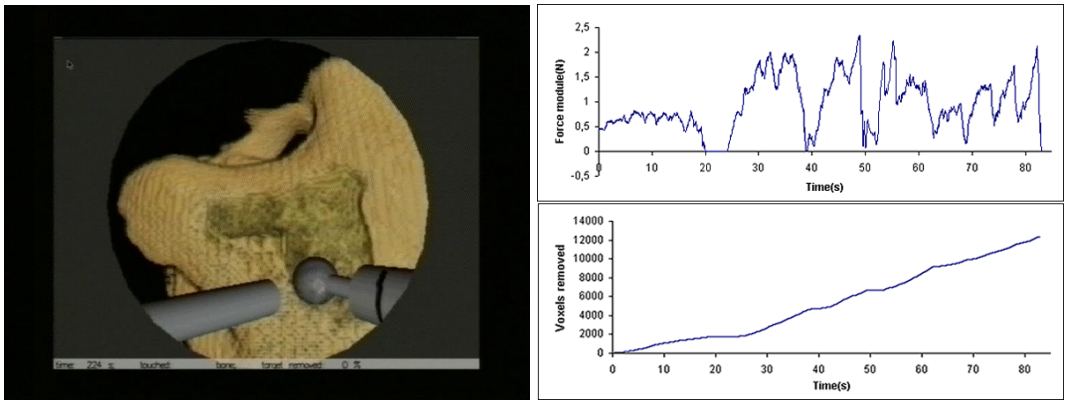


Figure 4.5: Mastoid cortex removal: Left: a snapshot of the simulator display during this operation. Right: plots of the force modulus and of the bone removal vs time. In this part we see low-frequency periodical variations and high force peak values.

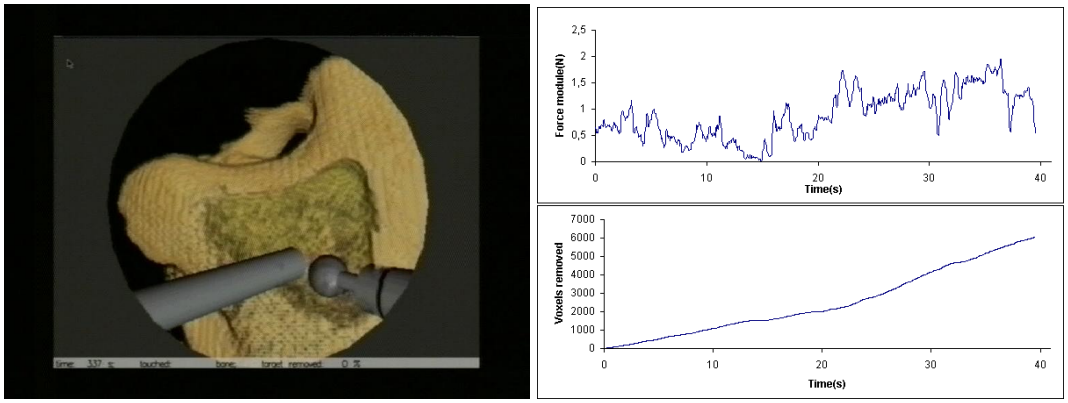


Figure 4.6: Cavity saucerization: Left: a snapshot of the simulator display during this operation. Right: plots of the force modulus and of the bone removal vs time.In this part we see only high frequency variations and high force peak values.

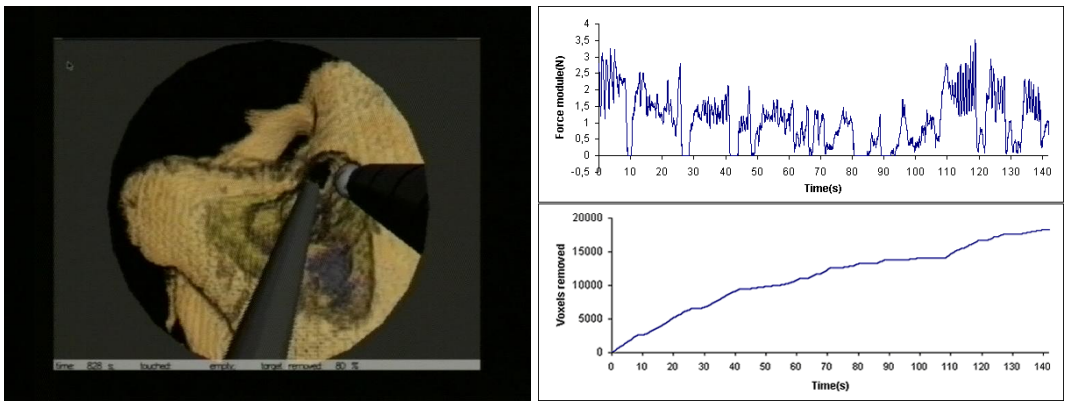


Figure 4.7: The mastoid antrum is the principal structure highlighted during this step. Left: a snapshot of the simulator display during this operation. Right: plots of the force modulus and of the bone removal vs time. In this part we see only high frequency variations and high force peak values.

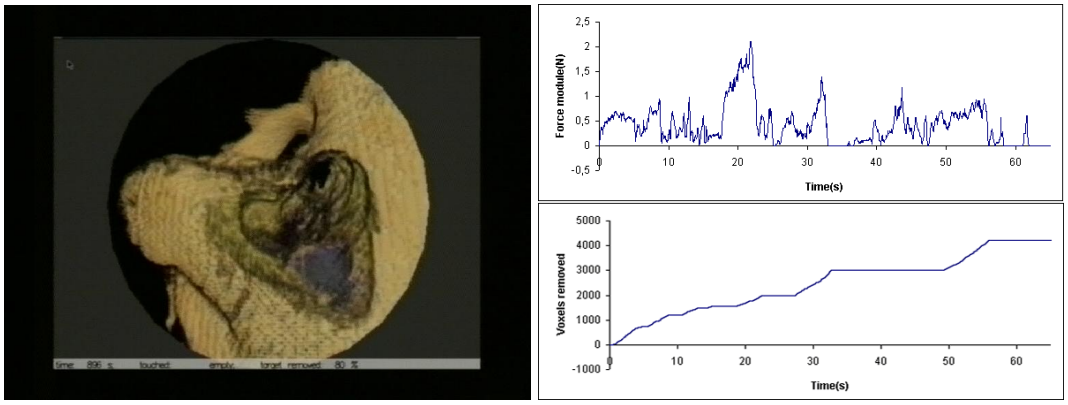


Figure 4.8: Identification of the mastoid portion of the facial nerve. Left: a snapshot of the simulator display during this operation. Right: plots of the force modulus and of the bone removal vs time. In this part we see only high frequency variations and high force peak values.

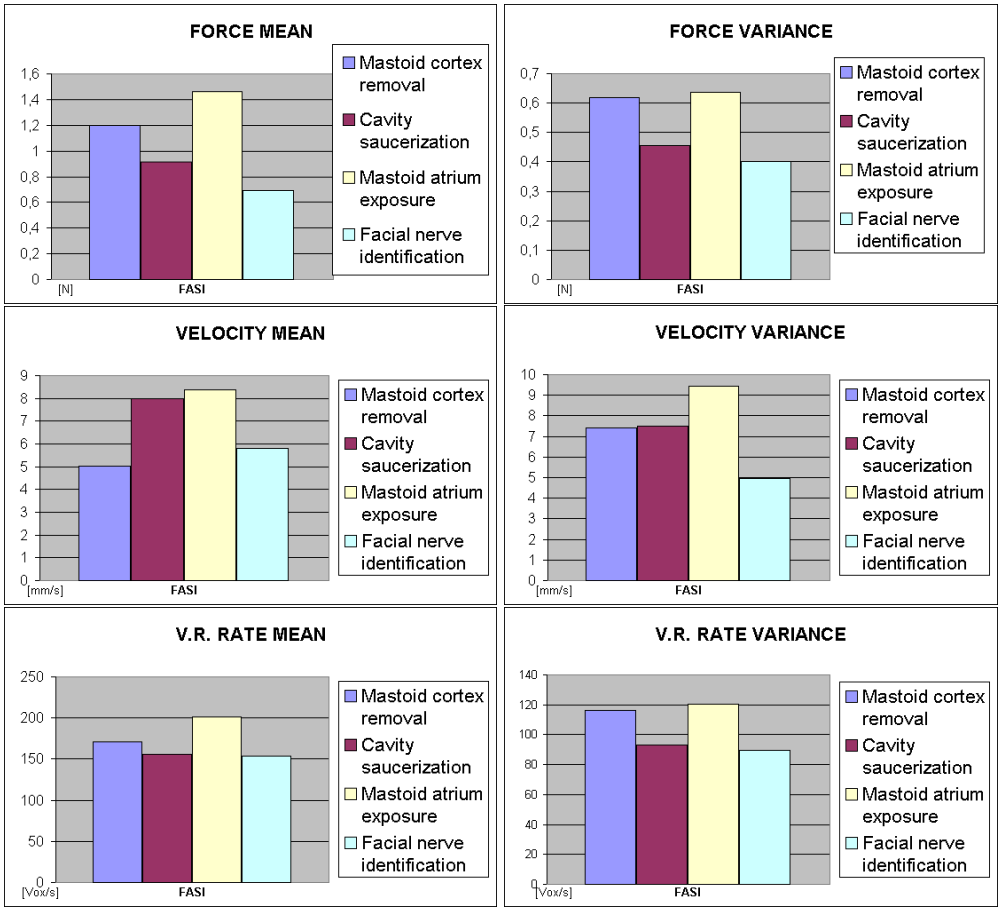


Figure 4.9: Average value and variance of the force modulus, velocity and bone voxels removed during the four mastoidectomy phases considered

Bibliography

[AGG+02a] Marco Agus, Andrea Giachetti, Enrico Gobbetti, Gianluigi Zanetti, Nigel W. John, and Robert J. Stone. Mastoidectomy simulation with combined visual and haptic feedback. In J. D. Westwood, H. M. Hoffmann, G. T. Mogel, and D. Stredney, editors, *Medicine Meets Virtual Reality 2002*, January 2002.

[AGG+02b] Marco Agus, Andrea Giachetti, Enrico Gobbetti, Gianluigi Zanetti, Nigel W. John, and Robert J. Stone. Mastoidectomy simulation with combined visual and haptic feedback. In J. D. Westwood, H. M. Hoffmann, G. T. Mogel, and D. Stredney, editors, *Medicine Meets Virtual Reality 2002*, pages 17–23. IOS Press, January 2002.

[AGG+02c] Marco Agus, Andrea Giachetti, Enrico Gobbetti, Gianluigi Zanetti, and Antonio Zorcolo. A multiprocessor decoupled system for the simulation of temporal bone surgery. *Computing and Visualization in Science*, 5(1), 2002.

[AGG+02d] Marco Agus, Andrea Giachetti, Enrico Gobbetti, Gianluigi Zanetti, and Antonio Zorcolo. Real-time haptic and visual simulation of bone dissection. In *IEEE Virtual Reality Conference, Conference*, Orlando, FL, USA, March 2002. IEEE Computer Society Press.

[AH99] R. Adams and B. Hannaford. Stable haptic interaction with virtual environments. *IEEE Transactions on Robotics and Automation*, 15(3):465–474, 1999.

[BSWS01] Jason Bryan, Don Stredney, Greg Wiet, and Dennis Sessanna. Virtual temporal bone dissection: A case study. In *IEEE Visualization*, pages 497–500, 2001.

[CCF94] Brian Cabral, Nancy Cam, and Jim Foran. Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. In Arie Kaufman and Wolfgang Krueger, editors, *1994 Symposium on Volume Visualization*, pages 91–98. ACM SIGGRAPH, October 1994. ISBN 0-89791-741-3.

[CN94] Timothy J. Cullip and Ulrich Neumann. Accelerating volume reconstruction with 3D texture hardware. Technical Report TR93-027, Department of Computer Science, University of North Carolina - Chapel Hill, May 1 1994.

[Col94] J. Colgate. Issues in the haptic display of tool use. In *Proceedings of ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pages 140–144, 1994.

[EKE01] K. Engel, M. Kraus, and T. Ertl. High quality pre-integrated volume rendering using hardware-accelerated pixel shading. In *EuroGraphics/SIGGRAPH Workshop on Graphics Hardware*, 2001.

[ESJ97] R.E. Ellis, N. Sarkar, and M.A. Jenkins. Numerical methods for the force reflection of contact. *ASME Transactions on Dynamic Systems, Modeling, and Control*, 119(4):768–774, 1997.

[GL94] S. Guan and R. G. Lipes. Innovative volume rendering using 3D texture mapping. In *Image Capture, Formatting and Display*, volume 2164 of *SPIE*. SPIE, 1994.

[GSMF97] S. Gibson, J. Samosky, A. Mor, and C. Fyock. Simulating arthroscopic knee surgery using volumetric object representations, real-time volume rendering and haptic feedback. *Lecture Notes in Computer Science*, 1205:369–374, 1997.

[GZ01] Enrico Gobbetti and Gianluigi Zanetti. Surgical simulation subsystem requirements and functional specification. Technical Report IERAPSI (IST-1999-12175) Deliverable D2(part2), IERAPSI Consortium, 2001.

[HD91] R. Held and N. Durlach. Telepresence, time delay, and adaptation. In Stephen R. Ellis, editor, *Pictorial Communication in Real and Virtual Environments*. Taylor and Francis, 1991.

[ISO99] ISO. *International Standard ISO 13407, Human-Centered Design Processes for Interactive Systems*, 1999.

[JTP+01] N. W. John, N. Thacker, M. Pokric, A. Jackson, G. Zanetti, E. Gobbetti, A. Giachetti, R. J. Stone, J. Campos, A. Emmen, A. Schwerdtner, E. Neri, S. Sellari Franceschini, and F. Rubio. An integrated simulator for surgery of the petrous bone. In J. D. Westwood, editor, *Medicine Meets Virtual Reality 2001*, pages 218–224, Amsterdam, The Netherlands, January 2001. IOS Press.

[Kil00] Mark J. Kilgard. A practical and robust bump-mapping technique for todays gpus. Technical report, NVIDIA Corporation, July 2000.

[Kul96] Todd Kulick. Building an opengl volume renderer. SGI Dev. News, 1996.

[LL86] L. Landau and E. Lifshitz. *Theory of elasticity*. Pergamon Press, 1986.

[LM93] X. Li and J.M. Moshell. Modeling soil: Realtime dynamic models for soil slippage and manipulation. In *Computer Graphics Proceedings, Annual Conference Series*, pages 361–368, 1993.

[Max95] Nelson Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, June 1995.

[MSG95] Tomasz Mazuryk, Dieter Schmalstieg, and Michael Gervautz. Zoom rendering: Improving 3-D rendering performance with 2-D operations. Technical Report CG, Institute of Computer Graphics, Vienna University of Technology, 1995.

[MZ92] M. McKenna and D. Zeltzer. Three dimensional visual display systems for virtual environments. *Presence*, 1(4):421–458, 1992.

[RS99] J.K. Hodgins R.W. Sumner, J.F. O'Brien. Animating sand, mud and snow. *Computer Graphics Forum*, 18:1, 1999.

[RSEB+00] C. Rezk-Salama, K. Engel, M. Bauer, G. Greiner, and T. Ertl. Interactive volume rendering on standard PC graphics hardware using multi-textures and multi-stage rasterization. In Stephan N. Spencer, editor, *Proceedings of the 2000 SIGGRAPH/EUROGRAPHICS Workshop on Graphics Hardware*, pages 109–118, N. Y., August 21–22 2000. ACM Press.

[Sto01] Bob Stone. A human-centred definition of surgical procedures. Technical Report IERAPSI (IST-1999-12175) Deliverable D2(part1), IERAPSI Consortium, 2001.

[VK96] Allen Van Gelder and Kwansik Kim. Direct volume rendering with shading via three-dimensional textures. In *1996 Volume Visualization Symposium*, pages 23–30. IEEE, October 1996. ISBN 0-89791-741-3.

[WBS+00] G. Wiet, J. Bryan, D. Sessanna, D. Streadney, P. Schmalbrock, and B. Welling. Virtual temporal bone dissection simulation. In J. D. Westwood, editor, *Medicine Meets Virtual Reality 2000*, pages 378–384, Amsterdam, The Netherlands, January 2000. IOS Press.

[WE98] Rüdiger Westermann and Thomas Ertl. Efficiently using graphics hardware in volume rendering applications. In Michael Cohen, editor, *SIGGRAPH 98 Conference Proceedings*, Annual Conference Series, pages 169–178. ACM SIGGRAPH, Addison Wesley, July 1998. ISBN 0-89791-999-8.

[YJN+95] Christine Youngblut, Rob E. Johnson, Sarah H. Nash, Ruth A. Wienclaw, and Craig A. Will. Review of virtual environment interface technology. IDA Paper P-3186, Institute for Defense Analysesc, March 1995.