



Project Number	IST-1999-12175
Project Title	IERAPSI

Deliverable Number	<b>D4.1</b>	
Deliverable Title	<b>Surgical Simulation Software Kernel</b>	
Version Number	<b>1.0</b>	
Nature	<b>Report</b>	
Dissemination Level	<b>Internal</b>	
Contractual Date of Delivery	<b>PM 24</b>	
Actual Date of Delivery	<b>PM 24</b>	
Primary Authors	<b>Enrico Gobetti</b>	<b>CRS4</b>
	<b>Gianluigi Zanetti</b>	<b>CRS4</b>
Other Contributors	<b>Marco Agus</b>	<b>CRS4</b>
	<b>Andrea Giachetti</b>	<b>CRS4</b>
	<b>Antonio Zorcolo</b>	<b>CRS4</b>



# Contents

<b>Executive Abstract</b>	<b>v</b>
<b>Synopsis</b>	<b>1</b>
<b>1 General background and context</b>	<b>3</b>
<b>2 Surgical simulation software kernel architecture</b>	<b>7</b>
2.1 The decoupled simulation model . . . . .	7
2.2 Bone–burr interaction model . . . . .	9
2.2.1 Continuum description . . . . .	10
2.2.2 Discretized description . . . . .	12
2.2.3 Sample–Estimate–Hold Interface . . . . .	13
2.3 Real–time visual rendering . . . . .	14
2.3.1 Shaded direct volume rendering of dynamic volumes . . . . .	14
2.3.2 Reducing fill-rate bottleneck . . . . .	17
<b>3 Hardware System Configuration</b>	<b>19</b>
<b>4 System Technical evaluation</b>	<b>21</b>
4.1 General system performance . . . . .	21
4.2 Force Evaluation . . . . .	21
4.3 Bone erosion . . . . .	22
<b>References</b>	<b>23</b>



# Executive Abstract

This Report has been prepared in fulfilment of **Deliverable D4.1**, required as an intermediate 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 **D4.1** relates to the **Surgical simulation software kernel**, the first of the two main expected results of Work Package 4.

The “Surgical simulation software kernel” will be used as the foundation upon which task T4.6 “Surgical simulator prototypes” will build deliverable **D4.2, Petrous bone surgical simulation platform**.

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 “Surgical simulation software kernel”, deliverable D4.1.
- 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 PIII/600 MHz with 256 MB PC100 RAM for to control the haptic devices; a dual-processor PIII/600 MHz with 512 MB PC800 RAM and a NVIDIA GeForce 2 GTS 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 first results of the system technical evaluation. The main finding is that the surgical simulation software kernel architecture respects

the requirements listed in D2 and provides a solid foundation to build the deliverable D4.2, “Petrous bone surgical simulation platform”.

The report concludes with a bibliography of cited reference work.

## **Copyright Notice**

The IERAPSI Project (an Integrated Environment for the Rehearsal and Planning of Surgical Interventions) is a collaboration between the University of Manchester, CRS4, the University of Dresden, University College London, the University of Pisa, Virtual Presence Ltd., Genias Benelux b.v. and CS-SI. The project is managed by the University of Manchester and is funded by the European Community under the IST Project IST-1999-12175.

# Synopsis

## Purpose of the Document

This Report has been prepared in fulfilment of Deliverable D4.1, required as an intermediate 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 **D4.1** relates to the **Surgical simulation software kernel**, the first of the two main expected results of Work Package 4.

The “Surgical simulation software kernel” will be used as the foundation upon which T4.6 “Surgical simulator prototypes” will build deliverable D4.2 “Petrous bone surgical simulation platform”.

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

## Structure of the Document

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 “Surgical simulation software kernel”, deliverable D4.1.
- 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.

- Section 4 summarizes the first results of the system technical evaluation. The main finding is that the surgical simulation software kernel architecture respects the requirements listed in D2 and provides a solid foundation to build the deliverable D4.2, “Petrous bone surgical simulation platform”.

The report concludes with a bibliography of cited reference work.

# 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<sup>+</sup>02]. 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.

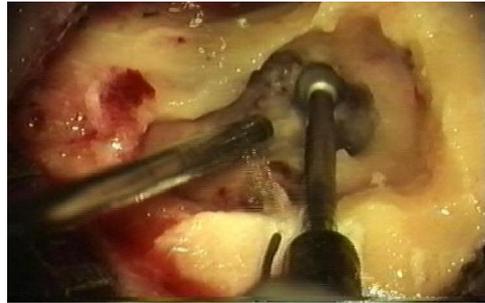
In the typical mastoidectomy surgical setup, Fig. 1.1(a), the ENT surgeon looks at the region of interest through a stereoscopic microscope and holds in his hands a high speed burr and a sucker. These tools are used, respectively, to cut the bone and to remove water (used to cool the burr bit) and bone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

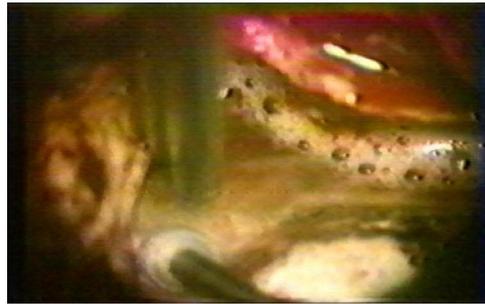
As for the visual effect of the drill on the surface of the bone, the task analysis highlighted that the graphical process must simulate drill site obscuration by bone dust paste, because its absence would reduce the importance placed by a trainee on the need for regular irrigation and suction. Realistic 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<sup>+</sup>02]. 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

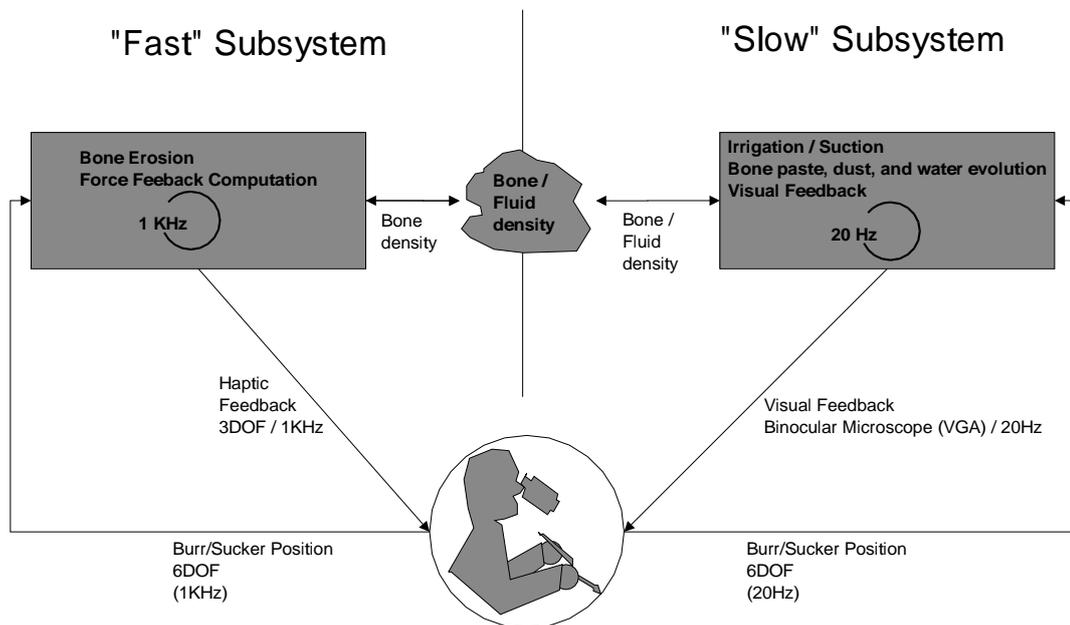


Figure 2.3: **Logical system decomposition.** The system is divided in a "fast" subsystem, responsible for the high frequency tasks (surgical instrument tracking, force feedback computation, bone erosion), and a "slow" subsystem, essentially dedicated to the production of data for visual feedback.

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.

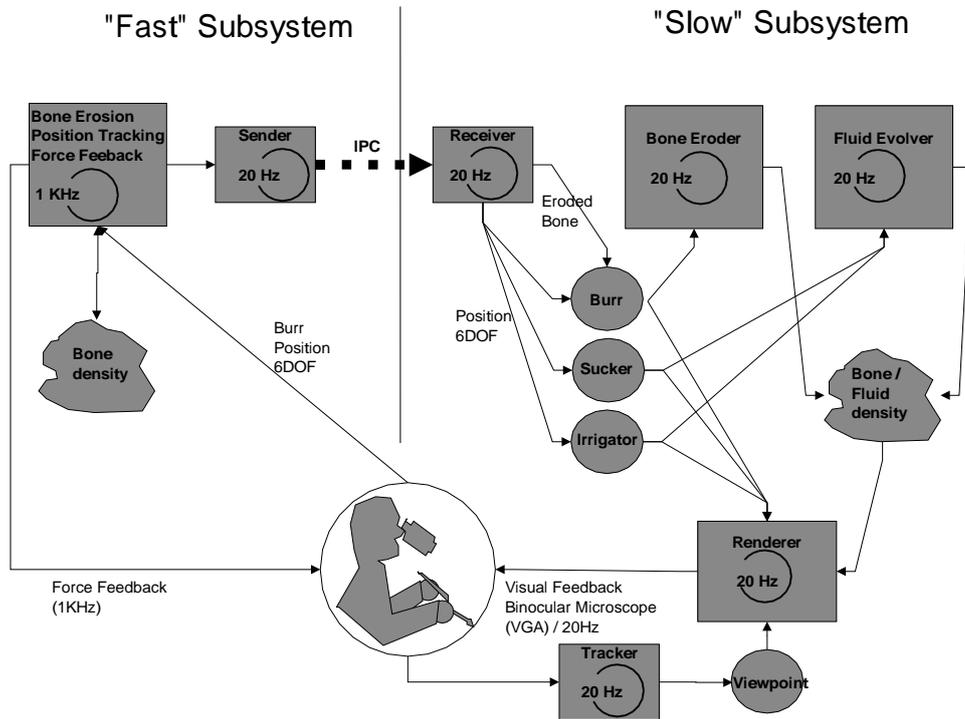


Figure 2.5: **Decoupled simulation architecture.** The system uses a volumetric approach, with the initial configuration of the model directly derived from patient CT data. The data is initially replicated on the two machines. The first machine is dedicated to the high-frequency tasks: haptic device 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 Bone–burr interaction model

A detailed mechanical description of a rotating burr cutting bone is complicated because it involves tracking 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.

To circumvent these complications, we have divided the cutting process in two successive steps. The first estimates the bone material deformation and the resulting elastic forces, given the relative position of the burr with respect to the bone. The second estimates the local rate of cutting of the bone by using a – postulated – energy

balance between the mechanical work performed by the burr motor and the energy needed to cut the bone, that it is assumed to be proportional to the bone mass removed.

We will first describe this approach on a continuum model and then specialize the results to a discretized voxel grid.

## 2.2.1 Continuum description

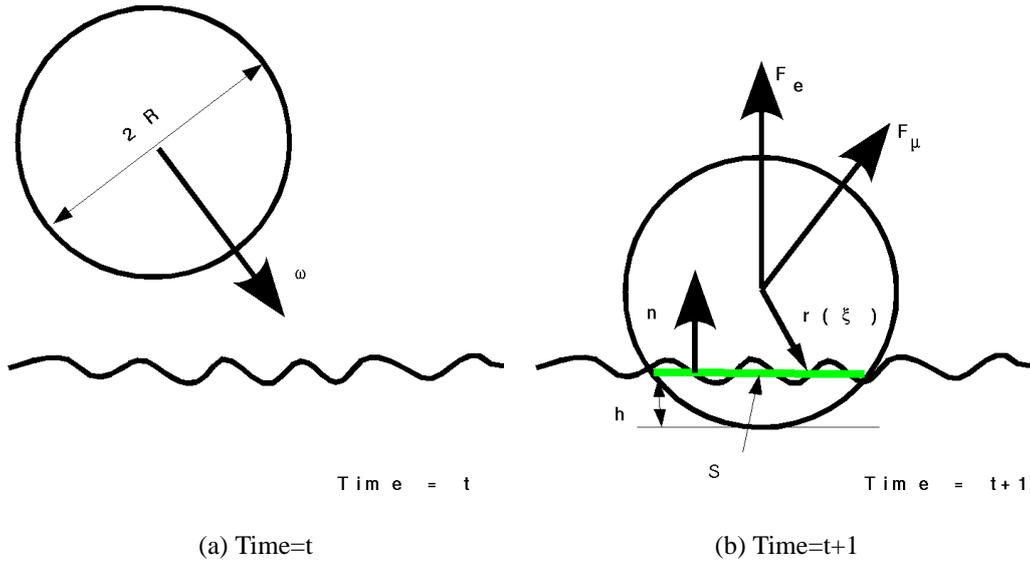


Figure 2.6: The impact of burr on bone. Here we represent two successive instants, at time  $t$  and  $t + 1$ , of an idealized version of a surgeon burr. The burr has a spherical bit, of radius  $R$ , that is rotating with angular velocity  $\vec{\omega}$ . The surface  $S$  is the effective “contact surface” between the burr and the bone.

In figure 2.6 we represent two successive instants, at time  $t$  and  $t + 1$ , of an idealized version of a surgeon burr. The burr has a spherical bit, of radius  $R$ , that is rotating with angular velocity  $\vec{\omega}$ . At time step  $t$  the burr is just outside the bone material, while at the next time step it is intersecting the bone surface. In the following, we will refer to the sphere representing the burr bit as  $B$ , and to the “contact surface” between the burr and the bone as  $S$ .

All the relevant geometrical information is contained in the volumetric distribution of the bone material. We use a characteristic function  $\chi(\vec{r})$  to indicate the presence/absence of bone, where  $\vec{r}$  is measured from the center of  $B$ . The first two moments of  $\chi$ , restricted to the region contained in  $B$  are, respectively,

$$M = \int_{r < R} d^3r \chi(\vec{r}), \quad (2.1)$$

$$\vec{M}_1 = \int_{r < R} d^3r \chi(\vec{r}) \vec{r}. \quad (2.2)$$

We can now estimate the normal direction,  $\hat{\mathbf{n}}$ , to  $S$ , as  $\hat{\mathbf{n}} = -\vec{M}_1/|M_1|$  and the “thickness”  $h$  of  $B$  immersed in the bone, by solving  $M = \pi h^2(R - \frac{h}{3})$ . We can now derive, assuming that  $\frac{h}{R} \ll 1$ , and using Hertz’s contact theory [LL86], an expression for the total force,  $\vec{F}_e$ , exerted on the burr by the elastic deformation of the bone:

$$\vec{F}_e = C_1 R^2 \left(\frac{h}{R}\right)^{\frac{3}{2}} \hat{\mathbf{n}}, \quad (2.3)$$

where  $C_1$  is a dimensional constant, that describes the elastic properties of the material. Moreover, we can give an expression for the pressure,  $\vec{P}(\vec{\xi})$ , exerted by the burr on the point  $\vec{\xi}$  of  $S$ :

$$\vec{P}(\vec{\xi}) = -\frac{3}{2\pi a^2} \sqrt{1 - \frac{|\vec{\xi}|^2}{a^2}} \vec{F}_e, \quad (2.4)$$

where  $\vec{\xi}$  is measured from the center of  $S$ , see fig. 2.6(b), and  $a$  is the radius of the contact region. In Hertz’s contact theory,  $a$  can be estimated as

$$a = (C_1 R)^{\frac{1}{3}} F_e^{\frac{1}{3}}. \quad (2.5)$$

From equation 2.4, we can estimate the frictional force,  $\vec{F}_\mu$ , that the bone will oppose to the burr rotation:

$$\vec{F}_\mu = \mu \int_{\xi < a} d\sigma P(\vec{\xi}) \frac{\vec{r}(\vec{\xi}) \times \vec{\omega}}{|\vec{r}(\vec{\xi})| |\vec{\omega}|}, \quad (2.6)$$

where  $\mu$  is a friction coefficient, that links the frictional forces for unit area to the locally exerted pressure.

The total force that should be returned by the haptic feedback device is, therefore,  $\vec{F}_T = \vec{F}_e + \vec{F}_\mu$ .

We model the cutting of the burr by assuming that all the power spent by working against the frictional forces on a “contact surface” element  $d\sigma$  goes toward the erosion of the bone material in contact with the surface. In other words, we equate

$$\mu P(\vec{\xi}) \omega r(\vec{\xi}) \left(1 - \left(\frac{\vec{r}(\vec{\xi}) \cdot \vec{\omega}}{|\vec{r}(\vec{\xi})| |\vec{\omega}|}\right)^2\right) d\sigma = \alpha \phi(\vec{\xi}) d\sigma, \quad (2.7)$$

where  $\alpha$  is a dimensional constant and  $\phi(\vec{\xi})$  is the mass flux at the contact surface point  $\vec{\xi}$ . Using the mass flux  $\phi$  one can update the position of the bone surface.

The formulas above have been written with the implicit assumption that the burr blades are very small with respect to the burr bit radius, and that their effect can be absorbed in the friction constant  $\mu$  and in the “erosion constant”  $\alpha$ . Even though this is, in general, false, and Hertz’s theory is, strictly speaking, only valid for small elastic deformations, this formulation provides a computationally tractable, robust, expression for the response forces that, at least in the limit of small  $h$ , is physically reasonable.

## 2.2.2 Discretized description

In the simulator, the bone distribution is only known at the level of a volumetric grid discretized in cubic voxels. Eqs. (2.1,2.2,2.6) need, therefore, to be translated and re-interpreted.

A direct translation will transform integrals in sums over the voxels that have non null intersection with  $B$ . The evaluation of each voxel contribution is computationally complex, since it requires to find the intersections between  $B$  and the cube defining the voxel. To simplify matters, we are approximating the voxels with spheres of the same volume, centered at the voxel center,  $\vec{c}_i$ , whit the origin at the center of  $B$ . The radius of the voxel spheres,  $\eta$ , is, therefore, defined by  $\frac{4}{3}\pi\eta^3 = \ell^3$ , where  $\ell$  is the length of the voxel side.

Using this approximation, it is trivial to derive simple formulas that express, in terms of the distance  $d = |\vec{c}_i|$ , the volume,  $\Delta V$ , of the intersection region; the area,  $\Delta\sigma$ , of the “intersection surface” and the actual distance,  $r$ , from the center of the intersection surface to the center of  $B$ .

$$\Delta v(d) = \frac{\pi}{12}(d^3 - 6(R^2 + \eta^2)d + 8(R^3 + \eta^3)) \quad (2.8)$$

$$- 3(\eta^2 - R^2)^2 \frac{1}{d} \quad (2.9)$$

$$\Delta\sigma(d) = \frac{\pi}{4}(2(\eta^2 + R^2) - d^2 - (\eta^2 - R^2)^2 \frac{1}{d^2}) \quad (2.10)$$

$$r(d) = \frac{1}{2}d + \frac{R^2 - \eta^2}{2} \frac{1}{d} \quad (2.11)$$

The required integrals then become

$$M^* = \sum_i \Delta V(|\vec{c}_i|) \chi_i \quad (2.12)$$

and

$$\vec{M}_1^* = \sum_i \Delta V(|\vec{c}_i|) \chi_i \frac{r_i}{d_i} \vec{c}_1. \quad (2.13)$$

To estimate the friction force,  $\vec{F}_\mu$  we convert the area integral (2.6) in

$$\vec{F}_\mu = \mu \sum_i \Delta\sigma(|\vec{c}_i|) P(\vec{\xi}_i) \frac{\vec{c}_i \times \vec{\omega}}{|\vec{c}_i| |\vec{\omega}|}, \text{ with } \vec{\xi}_i = \frac{r_i}{d_i} (\vec{c}_i - \frac{(\vec{\omega} \cdot \vec{c}_i)}{\omega^2} \vec{\omega}). \quad (2.14)$$

The power spent by the frictional forces on a voxel is then

$$\mu P(\xi_i) \omega r_i(\vec{\xi}_i) \left( 1 - \left( \frac{\vec{c}_i \cdot \vec{\omega}}{|\vec{c}_i| |\vec{\omega}|} \right)^2 \right) \Delta\sigma_i = \alpha \phi_i \Delta\sigma_i, \quad (2.15)$$

where  $\phi_i$  is the mass flux per unit surface coming out of voxel  $i$ , via surface  $\Delta\sigma_i$ . To evaluate  $P$  we use formula (2.4), where for  $a$  we use the “effective” radius of the contact surface  $a^* = \sqrt{2Rh - h^2}$ .

Using the fluxes  $\phi_i$  we can now erode the voxels in the intersection region. In our current implementation, we associate a 8 bit counter with each voxel, representing the voxel density, and decrease it by a value proportional to the “assumed” amount of removed mass,  $\Delta M_i = \Delta t \Delta \sigma \phi_i$ , where  $\Delta t$  is the time step of the simulation, and the mass,  $M_i$ , contained in the voxel  $i$ . The bone material in the temporal bone area has a morphological structure that ranges from compact bone, e.g., close to the outer skull surface, to a porous, “trabecular”, consistency. The porous scale ranges from few millimeters down to scales well beyond the resolution of the medical imaging devices. In our model, the subscale modeling of the trabecular structures is absorbed in a voxel dependent erosion constant  $\alpha$ .

As it was mentioned before, the burring of the bone produces dust that mixes with water in a paste, “mud”, clearly visible in figure 1.1(a). The paste material has a quite complex behavior, from sand-like to gel-like. The water paste mixture needs to be continuously removed, otherwise it can obscure the field of view as it is seen in figure 1.1(b). 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, since its “obscuring” effects constitute the principal cue to the user for the use of the sucker device.

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. Therefore, we are modeling the dust/fluid dynamics using what essentially amounts to an hybrid particles/sand pile model [RS99, LM93]. The dust/fluid system is fed by the burring but its dynamics does not influence the haptic force evaluation.

Figure 4.4 shows the beginning of a typical bone cutting sequence performed in the mastoid region.

### 2.2.3 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].

## 2.3 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<sup>+</sup>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.

### 2.3.1 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<sup>+</sup>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:

$$I(x, y, z) = (c_a + c_d \left| \vec{\nabla} k_m^\alpha(x, y, z) \cdot \vec{l} \right|) \cdot k_m(x, y, z) \quad (2.16)$$

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<sup>+</sup>01, AGG<sup>+</sup>02]. The shading formula becomes, using a forward difference approximation of the gradient:

$$\frac{\Delta k_m^\alpha}{\Delta z} = \left| \frac{k_m^\alpha(x, y, z + \Delta z) - k_m^\alpha(x, y, z)}{\Delta z} \right| \quad (2.17)$$

$$I(x, y, z) = (c_a + c_d \frac{\Delta k_m^\alpha}{\Delta z}) \cdot k_m(x, y, z) \quad (2.18)$$

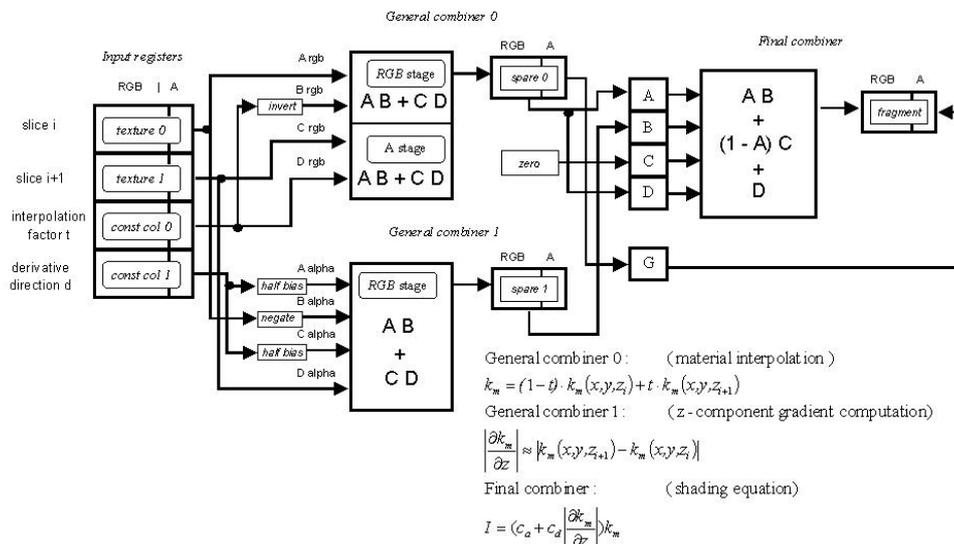


Figure 2.7: **OpenGL combiner setup.** The final combiner blends the interpolated slice value from the first combiner with the shading value from the second combiner.

This equation can be implemented in the graphics hardware by programming the register combiners (see figure 2.7), 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 2.7 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 2.17.

This procedure is extremely efficient, since all the computation is 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 transferred to texture memory at over 60 fps. Only two slices need to be present in texture memory at the same time.

### **2.3.2 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 PIII/600 MHz with 256 MB PC100 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/600 MHz with 512 MB PC800 RAM and a NVIDIA GeForce 2 GTS 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 PIII/600 MHz with 256 MB PC100 RAM (haptic loop) and a dual-processor PIII/600 MHz with 512 MB PC800 RAM and a NVIDIA GeForce 2 GTS.

# Chapter 4

## System Technical 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  $256 \times 256 \times 128$  cubical voxels (0.3 mm side) obtained by manually adding soft-tissue information to an high resolution CT scan. We are in the process of adapting the system to the direct use of dataset obtained by applying probability maps methods [PTSJ01] for automatic multi-dimensional medical image segmentation.

The force-feedback loop is running at 1 KHz using a  $5 \times 5 \times 5$  grid around the tip of the instruments 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  $800 \times 600$  window with 16 bit color and 2X zoom rendering.

### 4.2 Force Evaluation

Figure 4.1 shows the reaction of the virtual bone against burr penetration. The computations are done in absence of erosion,  $\alpha = \infty$ , and using the actual force evaluation kernel of the force-feedback loop.

Figure 4.1(a) illustrates the “elastic” response of the material, measured in units of  $C_1 R^2$ , as a function of the burr tip penetration depth measured in units of the burr bit radius  $R$ . Figure 4.1(b) illustrates the “frictional” response of the material, with  $\mu = 1/2$  and for different angles  $\theta$ ,  $\theta = 30^\circ, 60^\circ, 90^\circ$ , between the surface normal and

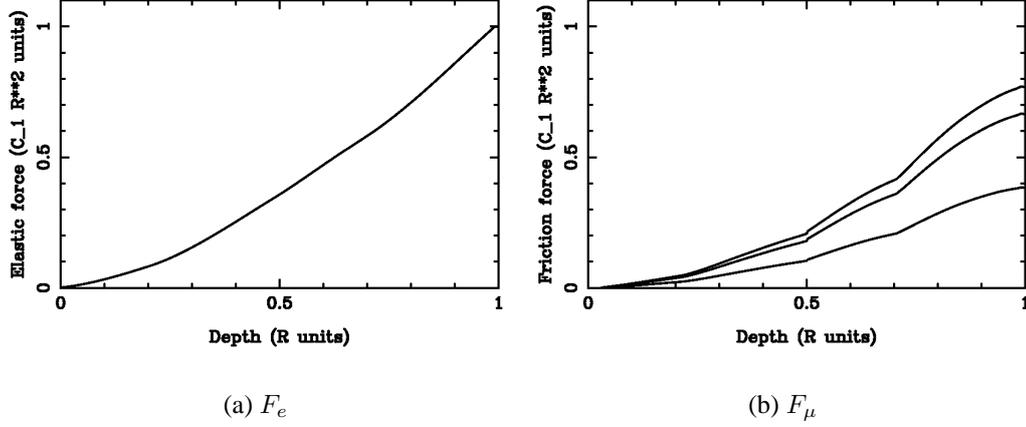


Figure 4.1: Virtual bone reaction against burr penetration. The computations are done in absence of erosion,  $\alpha = \infty$ , using the actual force evaluation kernel of the force–feedback loop. In (a) we show the “elastic” response of the material, measured in units of  $C_1 R^2$ , as a function of the burr tip penetration depth in units of the burr bit radius  $R$ . Fig. (b) illustrates the “frictional” response of the material, with  $\mu = 1/2$  and for different angles  $\theta$ ,  $\theta = 30^\circ, 60^\circ, 90^\circ$ , between the surface normal and  $\hat{\omega}$ . The strength of  $F_\mu$  increases for increasing  $\sin(\theta)$ . The knees in the  $F_\mu$  curves correspond to the intersection of the burr bit with a deeper bone voxel layer.

$\hat{\omega}$ . The strength of  $F_\mu$  increases for increasing  $\sin(\theta)$ . The knees in the  $F_\mu$  curves correspond to the intersection of the burr bit with a deeper bone voxel layer.

Figure 4.2 shows the reaction of the virtual bone, again in runs with  $\alpha = \infty$ , to a sliding motion of the burr bit, immersed at a depth of  $R/4$ , over a flat bone surface. Fig. 4.2(a,b) show, respectively, the “elastic” and the “frictional” force response of the material, measured in units of  $C_1 R^2$ , as a function of the distance traveled along the plane measured in  $R$  units. The pair of curves in each figure correspond to a sliding motion over a bone surface aligned along, respectively, one of the voxel discretization axis, and a plane with normal  $[0, \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}]$ . The fluctuations in the force values are due to the “voxel sphere” approximation used to compute  $F$ . The difference in the wavelength of the fluctuations is a factor of  $\sqrt{2}$  as expected.

### 4.3 Bone erosion

Figure 4.3 illustrates a “free–hand” experiment where bone is eroded by a polishing movement. The movement is similar to the one described in the previous subsection, with a sliding speed of about 10mm/sec, and  $\alpha = 3.1 \times 10^6 \text{mm}^2/\text{sec}^2$ . Figure 4.3(a) shows the depth of the burr below the surface level as a function of time, while fig. 4.3(b) reports the components of the force contributions and the total force applied to the haptic display during the movement.

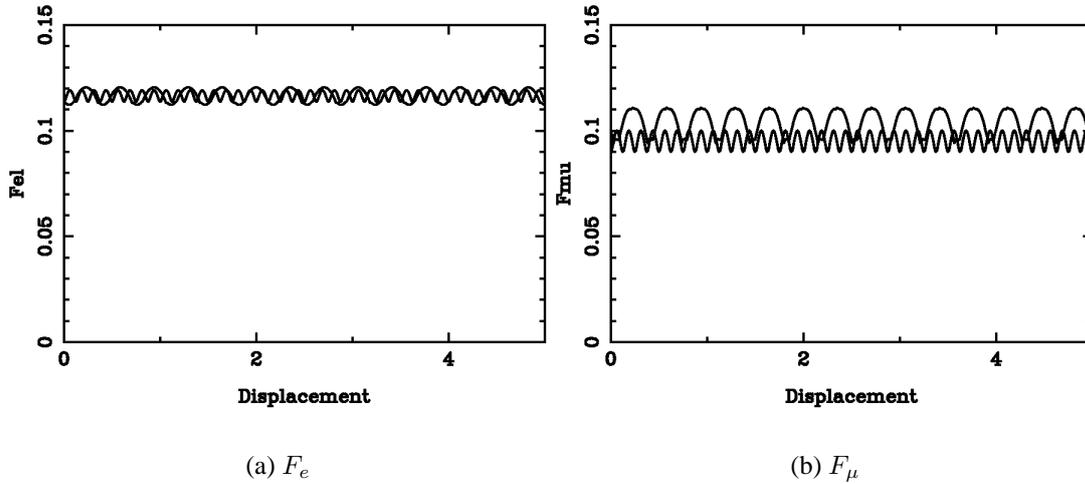
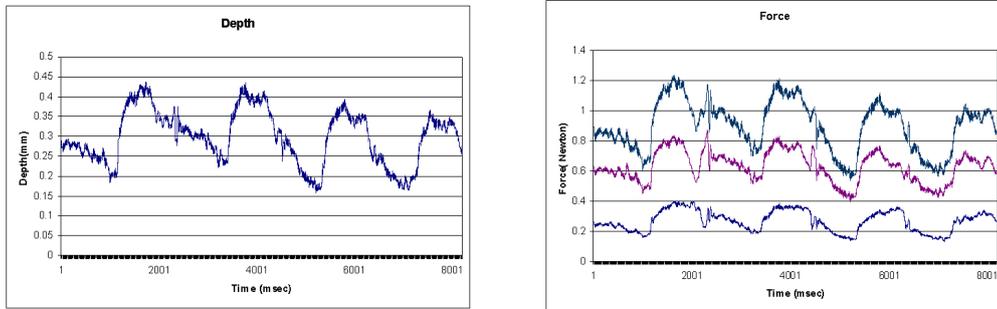


Figure 4.2: Sliding motion, constrained experiment. The reaction of the flat surface of virtual bone to the sliding motion of a burr bit immersed at a depth of  $R/4$ . Fig. (a,b) show, respectively, the “elastic” and the “frictional” force response of the material, measured in units of  $C_1 R^2$ , as a function of the distance traveled along the plane measured in  $R$  units. The pair of curves in each figure correspond to a sliding motion over a bone surface aligned along, respectively, one of the voxel discretization axis, and a plane with normal  $[0, \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}]$ . The fluctuations in the force values are due to the “voxel sphere” approximation used to compute  $F$ . The difference in the wavelength of the fluctuations is a factor of  $\sqrt{2}$  as expected.

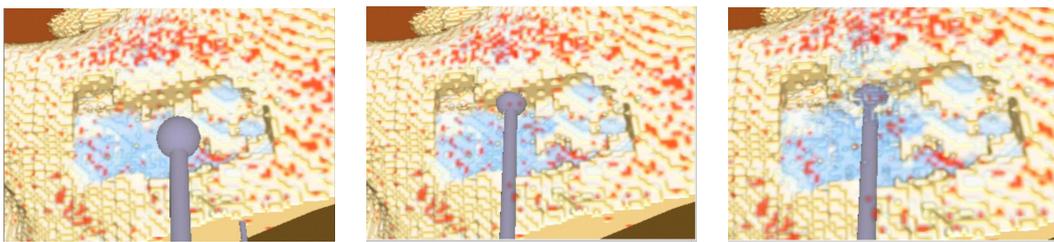
We have gathered initial feedback about the prototype system from specialist surgeons from the University of Pisa that are collaborating to this research. Subjective input is being used to tune the parameters that control force feedback. The overall realism of the simulation is considered sufficient for training purposes. Fig. 4.4 shows a typical erosion sequence. A demonstration movie is available on the IERAPSI project web site [ABG<sup>+</sup>01].



(a) Depth

(b) Forces

Figure 4.3: Bone erosion, polishing movement. A “free-hand” experiment where bone is eroded by a polishing movement. The sliding speed is about 10mm/sec, and  $\alpha = 3.1 \times 10^6 \text{ mm}^2/\text{sec}^2$ . Fig. (a) shows the depth of the burr below the surface level as a function of time. Fig. (b) reports the components of the force contributions and the total force applied to the haptic display during the movement. The lower line is the friction force  $\vec{F}_\mu$ , the middle line is the elastic force  $\vec{F}_{el}$ , and the upper line is the total force  $\vec{F}_{tot}$ .



(a)

(b)

(c)

Figure 4.4: A virtual burring sequence. Here we show a typical bone cutting sequence performed in the mastoid region. The accumulation of debris, and its masking effects, is clearly visible.

# Bibliography

- [ABG<sup>+</sup>01] Marco Agus, Fabio Bettio, Andrea Giachetti, Enrico Gobbetti, Gianluigi Zanetti, and Antonio Zorcolo. Real-time haptic and visual simulation of bone dissection. Video demonstration, <http://www.crs4.it/ierapsi>, 2001.
- [AGG<sup>+</sup>02] 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.
- [AH99] R. Adams and B. Hannaford. Stable haptic interaction with virtual environments. *IEEE Transactions on Robotics and Automation*, 15(3):465–474, 1999.
- [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<sup>+</sup>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.
- [PTSJ01] Maja Pokric, Neil Thacker, Marietta L.J. Scott, and Alan Jackson. The importance of partial voluming in multi-dimensional medical image segmentation. *Lecture Notes in Computer Science*, 2208:1293–1294, 2001.
- [RS99] J.K. Hodgins R.W. Sumner, J.F. O’Brien. Animating sand, mud and snow. *Computer Graphics Forum*, 18:1, 1999.
- [RSEB<sup>+</sup>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.
- [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<sup>+</sup>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.