
Doctoral Dissertations

Student Theses and Dissertations

Spring 2008

Modeling and rendering for development of a virtual bone surgery system

Qiang Niu

Follow this and additional works at: https://scholarsmine.mst.edu/doctoral_dissertations

 Part of the [Mechanical Engineering Commons](#)

Department: Mechanical and Aerospace Engineering

Recommended Citation

Niu, Qiang, "Modeling and rendering for development of a virtual bone surgery system" (2008). *Doctoral Dissertations*. 2213.

https://scholarsmine.mst.edu/doctoral_dissertations/2213

This thesis is brought to you by Scholars' Mine, a service of the Missouri S&T Library and Learning Resources. This work is protected by U. S. Copyright Law. Unauthorized use including reproduction for redistribution requires the permission of the copyright holder. For more information, please contact scholarsmine@mst.edu.

**MODELING AND RENDERING FOR DEVELOPMENT OF
A VIRTUAL BONE SURGERY SYSTEM**

by

QIANG NIU

A DISSERTATION

**Presented to the Graduate Faculty of the
MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY**

In Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

in

MECHANICAL ENGINEERING

2008

Approved by:

Dr. M. C. Leu, Advisor

Dr. F. W. Liou

Dr. K. Krishnamurthy

Dr. X. Du

Dr. M. G. Hilgers

© 2008
Qiang Niu
All Rights Reserved

ABSTRACT

A virtual bone surgery system is developed to provide the potential of a realistic, safe, and controllable environment for surgical education. It can be used for training in orthopedic surgery, as well as for planning and rehearsal of bone surgery procedures.

This dissertation presents the techniques and methods of modeling and rendering for the developed virtual bone surgery system. Image processing and a region growing algorithm are used to systematically and efficiently extract different bone structures from CT scan data. Based on the processed CT data, the bone geometry is represented with volumetric modeling. The volumetric data is then used to generate polygonal faces through the marching cube algorithm. Bounding volume and quadtree-based adaptive subdivision techniques are developed to handle the large data set in order to achieve the real-time simulation and visualization required for virtual bone surgery. Implicit surface is used for surgical tool modeling and representation, and material removal simulation is achieved by continuously performing Boolean subtraction of the surgical tool model from the bone model. To make the virtual surgery system more intuitive and interactive, multi-point collision detection and balance point based force calculation are developed for haptic rendering. Sound analysis is also conducted to characterize the bone drilling sound. Spectral modeling synthesis method is applied to obtain the sinusoidal model and residual model, which are then used for auditory rendering in the bone surgery system.

Using the developed system, the user can perform virtual bone surgery by simultaneously “seeing” bone material removal through a graphic display device, “feeling” the force via a haptic device, and “hearing” the sound of tool-bone interaction.

ACKNOWLEDGMENTS

First of all, I would like to express my deepest gratitude to my advisor, Dr. Ming C. Leu, for his remarkable guidance and constant support throughout my study and research. During the course of this work, Dr. Leu has given me innumerable timely, patient, and valuable suggestions and advice.

I am also grateful to the members of my dissertation committee, Dr. F. W. Liou, Dr. K. Krishnamurthy, Dr. X. Du, and Dr. M. G. Hilgers for their help, encouragement, and suggestions.

I also wish to thank everyone from the Virtual Reality and Rapid Prototyping Laboratory with whom I had pleasure to work.

Last but not least, I would like to thank my family for their support in my academic endeavors. Without their love, understanding, encouragement, and patience, I would never have reached this milestone in my life.

TABLE OF CONTENTS

	Page
ABSTRACT	iii
ACKNOWLEDGMENTS	iv
LIST OF ILLUSTRATIONS	viii
LIST OF TABLES	xi
SECTION	
1. INTRODUCTION	1
1.1. BACKGROUND	1
1.2. CURRENT STATE OF DIGITAL SURGERY	3
1.3. KEY TECHNOLOGIES IN BONE SURGERY SIMULATION	7
1.4. DISSERTATION LAYOUT	9
2. VIRTUAL BONE SURGERY SYSTEM OVERVIEW	11
2.1. OBJECTIVES AND CHALLENGES	11
2.2. HARDWARE AND SOFTWARE CONFIGURATION	12
2.3. VIRTUAL BONE SURGERY SIMULATION ARCHITECTURE	13
3. MEDICAL IMAGE PROCESSING AND SEGMENTATION	15
3.1. IMAGE PROCEDURES	15
3.2. IMAGE PROCESSING AND SEGMENTATION	17
4. GEOMETRIC MODELING AND GRAPHIC RENDERING	21
4.1. GEOMETRIC MODELING	21
4.2. GRAPHIC RENDERING	23
4.2.1. Surface Rendering	23
4.2.2. Volume Rendering	25
5. LARGE MEDICAL DATA MANAGEMENT	26
5.1. INTRODUCTION	26
5.2. DATA MANAGEMENT METHODS	28
5.2.1. Bounding Volume	28
5.2.2. Quadtree-Based Adaptive Subdivision	32
5.2.3. Data Structure	34

5.3. SOME RESULTS OF DATA MANAGEMENT	36
5.4. DATA AND MODEL UPDATE DURING TOOL-BONE INTERACTION .	38
5.5. ADAPTIVE RESOLUTION IMPROVEMENT	40
6. MODELING AND REPRESENTATION OF SURGICAL TOOLS	44
6.1. SURGICAL TOOL'S REPRESENTATION	44
6.2. VIRTUAL TOOL RELATED GEOMETRICAL CALCULATION	49
7. FORCE MODELING AND HAPTIC RENDERING.....	52
7.1. INTRODUCTION	52
7.1.1. Force Modeling.	53
7.1.2. Collision Detection and Force Generation.	57
7.2. MULTI-POINT COLLISION DETECTION	60
7.3. CUTTING FORCE CALCULATION.....	62
8. SOUND MODELING AND RENDERING	68
8.1. INTRODUCTION	68
8.2. REVIEW ON SOUND MODELING METHODS.....	69
8.2.1. Physical Modeling Synthesis.....	69
8.2.2. Spectral Modeling Synthesis.....	70
8.3. METHODS OVERVIEW FOR SOUND MODELING AND RENDERING .	72
8.4. SOUND ACQUISITION	74
8.5. SOUND ANALYSIS	78
8.5.1. Sound Analysis in Time Domain.	78
8.5.2. Drilling Sound Sources Analysis.	80
8.5.3. STFT, PSD, Spectrogram and WT.....	84
8.5.4. Sound Analysis Using STFT and WT.....	90
8.5.5. Spectral Subtraction.	102
8.6. SOUND MODELING	107
8.6.1. Sinusoidal Modeling.....	108
8.6.2. Residual Modeling.	113
8.6.3. Sound Synthesis.	113
8.6.4. Sound Modeling and Synthesis Results.	115
8.7. SOUND RENDERING.....	123

9. SYSTEM IMPLEMENTATION, INTEGRATION AND RESULTS	125
9.1. IMPLEMENTATION OVERVIEW	125
9.2. MULTI-THREADING WITH MULTI-RATE	128
9.3. GUI OF THE VIRTUAL BONE SURGERY SYSTEM.....	131
9.3.1. Main GUIs for Different Modules.....	131
9.3.2. Other GUIs for the Parameters Selection.	134
9.4. SIMULATED MATERIAL REMOVAL PROCESSES	135
10. CONCLUSIONS AND FUTURE WORK.....	139
10.1. CONCLUSIONS.....	139
10.2. FUTURE IMPROVEMENTS	140
APPENDICES.....	142
A. MAIN PROCEDURES OF KNEE SURGERY AND HIP SURGERY	142
B. FLYOUT TOOLBAR FOR VIRTUAL SURGICAL TOOLS.....	144
BIBLIOGRAPHY.....	146
VITA	155

LIST OF ILLUSTRATIONS

Figure	Page
1.1. Example orthopedic operations.....	2
1.2. Some examples of temporal bone surgery simulators	5
1.3. Typical machining operations in surgery of knee and hip bones.....	6
1.4. Schematic of a basic bone surgery simulation system.....	7
1.5. Key elements involved in bone surgery simulation.....	8
2.1. User interface of the virtual bone surgery system with the haptic device	12
2.2. System architecture of the virtual bone surgery simulation system for this research	14
3.1. The most commonly used medical imaging techniques	15
3.2. Example of image data before and after processing	17
3.3. Region growing image segmentation algorithm	19
4.1. A volume seen as a stack of images and a volume seen as a 3d lattice of voxels	21
4.2. The marching cube algorithm for surface rendering of voxel data.....	24
5.1. Octree representation	27
5.2. An example of bounding volume selection	29
5.3. Bounding volumes of a bone	30
5.4. Uniform division of a model into sub-volumes	32
5.5. An example of quadtree subdivision.....	34
5.6. Data structure for the presented method	35
5.7. Example of tool-bone interaction and active sub-volume queue.....	39
5.8. Voxel walkthrough during tool-bone interaction.....	40
5.9. Sub-volume level to sub-voxel level.....	41
5.10. Adaptive tool-guided resolution improvement methods.....	42
5.11. Examples of resolution improvement for graphic display	43
6.1. Three regions of implicit function	45
6.2. An example of reaming tool representation.....	46
6.3. Examples of sample points on the tool's surface.....	47
6.4. Guides modeling for total knee replacement	48
6.5. Some geometric models for cutting tools	48
6.6. Simplified calculation for the relationship between guide and tool	50

7.1. Structure of haptic rendering	52
7.2. Modeling force in drilling a long bone	54
7.3. Problems of penalty based haptic rendering	58
7.4. Haptic rendering by virtual proxy	59
7.5. An example of multi-point collision detection	62
7.6. An example of force calculation for two consecutive time steps	64
7.7. The simulated force profile for burring on cortical bone.....	65
7.8. The simulated force profile for milling on cortical bone.....	66
8.1. Block diagram of sound modeling and rendering.....	73
8.2. Schematic of experiment for sound acquisition.....	75
8.3. Experiment testing materials.....	77
8.4. PCM wave file format.....	79
8.5. Drilling on different materials in time domain	79
8.6. Different drilling states and transitions of drilling on cortical bone material in time domain	80
8.7. Schematic illustration of the major components of a hand-held power drill.....	83
8.8. Block diagram of power spectrum calculation	87
8.9. Multiple-level wavelet packet decomposition tree	89
8.10. Power spectral density of free-running drilling sound.....	91
8.11. Power spectrum of bone drilling sound average.....	93
8.12. Wavelet packet decomposition of different drilling sounds	95
8.13. Spectrogram of drilling sound for the 3 rd generation composite humerus.....	98
8.14. Wavelet analysis of drill sound on the 3 rd generation composite humerus.....	100
8.15. Wavelet analysis for the frequency bands of [0, 689] Hz and [5512, 6890] Hz....	101
8.16. Displacement vs. wavelet analysis for the frequency bands of [0, 689] Hz and [5512, 6890] Hz	102
8.17. Block diagram of spectral subtraction	103
8.18. Spectral subtraction result: comparison of bone drilling sound, drill free-running sound and drilling sound difference in frequency domain.....	106
8.19. Spectral subtraction result: drilling sound difference in time domain.....	106
8.20. Block diagram of spectral modeling synthesis	109
8.21. Temporary results of peak detection for four different frames of free-running sound	111
8.22. Original peak continuation algorithm	112

8.23. Block diagram of the spectral synthesis	114
8.24. Spectral modeling synthesis results shown in spectrograms	116
8.25. Residual spectrum and its line-segment approximation of free-running drilling ..	119
8.26. Comparisons between original and spectral synthesis sound for drill free-running sound	119
8.27. Comparisons between real bone drilling sound and synthesized sound for cortical bone drilling	121
8.28. Comparisons between real bone drilling sound and synthesized sound for cancellous bone drilling	122
8.29. DirectSound sound buffering process with bone surgery simulation	123
9.1. Major C++ class diagram in UML format	126
9.2. Block diagram of multi-threading in the virtual bone surgery simulation	129
9.3. Flowchart of multi-threading for simulation thread, haptic thread, graphics thread, and auditory thread	131
9.4. Graphic user interface of the image processing module	132
9.5. Graphic user interface of the simple training module	133
9.6. GUI dialog for the current tool's properties change	134
9.7. GUI dialog for sound control and display	135
9.8. Illustration of free drilling	136
9.9. Illustration of guided drilling	137
9.10. Some screen shots of the machining processes	138

LIST OF TABLES

Table	Page
3.1. HU values and different materials and tissue types	16
5.1. Comparisons among the original data, object bounding box (OBB) data, and sum of section bounding boxes (SBB Sum) data.....	36
5.2. Various ratios among the original data, object bounding box (OBB) data, and sum of section bounding boxes (SBB Sum) data.....	37
5.3. The numbers of sub-volumes for different numbers of subdivisions resulting from the uniform division and the quadtree subdivision and their ratio	37
5.4. Methods comparison for resolution improvement.....	43
8.1. Relationship between the vibration frequencies and their sources	84

1. INTRODUCTION

1.1. BACKGROUND

Becoming a skillful surgeon requires rigorous training and iterative practice. Traditional training and learning methods for surgeons are based upon the Halstedian apprenticeship model, i.e., “see one, do one, teach one”, which is almost 100 years old [Haluck et al., 2000]. Students often watch and perform operations on cadaveric or synthetic bones under the tutelage of experienced physicians before performing the procedure themselves under expert supervision. They need to learn and perform material removal operations such as drilling and burring, as shown in Figure 1.1. Mistakes can lead to irreparable defects to the bone and the surrounding soft tissue during such procedures, which can result in complications such as early loosening, mal-alignment, dislocation, altered gait, and leg length discrepancy [Conditt et al., 2003]. The current system of surgery education has many challenges in terms of flexibility, efficiency, cost, and safety. Additionally, as new types of operations are developed rapidly, practicing surgeons need more efficient methods of surgical skill education [Gorman et al., 2000].

Virtual Reality (VR) is one of the most active areas of research in computer simulation. Virtual reality systems use computers to create virtual environments that simulate real-world scenarios. Special devices such as head-mounted displays, haptic devices, and data gloves are used for interacting in virtual environments to give real-world like feedback to the user. The most important factor contributing to VR development has been the arrival of low-cost, industry-standard multimedia computers and high-performance graphic hardware. VR has been integrated into many aspects of modern society, such as engineering, architecture, entertainment.

The concept of developing and integrating computer-based simulation and training aids for surgery training began with VR simulators. VR techniques provide a realistic, safe, and controlled environment in which novice surgeons can practice surgical operations, allowing them to make mistakes without serious consequences. It promises to change the world of surgical training and practice. With a VR simulator, novice surgeons can practice and perfect their skills on simulated human models, and experienced

surgeons can use the simulator to plan surgical procedures. VR training also offers the possibility of providing a standardized performance evaluation for the trainees.



Figure 1.1. Example orthopedic operations: (a) bone burring for mastoidectomy surgery [Augus et al., 2002]; (b) drilling of the matatarsal phalanx in toe arthroplasty surgery (<http://www.orthosonics.com/>)

Bone surgery is one of the medical applications that can be simulated using VR technology. Some surgical simulation tools for orthopedic applications such as knee surgery exist, but most of them involve only soft tissues. Few have considered the simulation of cutting, sawing, burring, etc., which involve operating on bones, as well as on ligaments and muscles. Therefore, the development of a virtual bone surgery system would enable students in training to visualize surgical operations simulated with an added sense of touch. As the Minimally Invasive Surgery (MIS) takes a foot-hold in orthopedics, VR technology will become increasingly valuable for assisting actual surgery operations. In accordance with the MIS concept, surgical techniques are developed to sequentially reduce access to the surgical site (via smaller incisions), and instruments and implants are miniaturized to accommodate for these techniques. In such cases, surgical dexterity, bone preparation, and implant positioning become an increasingly less forgiving part of the operation. In order to assist surgeons in performing

the MIS process, it will be necessary to integrate VR models with images obtained during actual surgery operations to create the so-called Augmented Reality (AR) technology.

1.2. CURRENT STATE OF DIGITAL SURGERY

Previous research on surgery simulation has covered a wide range of operations. Some of the simulators were developed to provide a virtual reality environment as a training tool. The VRMedLab networked facility at the University of Illinois-Chicago [VRMedLab, 2003] was designed to provide an educational resource for otolaryngology surgeons, enabling them to visualize bone-encased structures within the temporal bone using interactive 3D visualization technology. The Ohio Supercomputer Center and the Ohio State University Hospital developed an endoscopic sinus-surgery simulator, which provided the capability of intuitive interaction with complex volume data and haptic feedback sensation [Edmond et al., 1997]. CathsimTM was an example of a commercially available training system for venipuncture [Barker, 1999]. Users of Cathsim could practice inserting a needle into a virtual vein in different scenarios. Røtnes et al. [2002] designed a coronary anastomosis simulator called SimMentorTM, which exploits a common geometric model to provide animated 3D visualization and interactive simulation during training sessions.

Some of the surgery simulations involved deformable geometric models in the software system. Delp et al. [1997] developed tissue cutting and bleeding models for this purpose. Bro-Nielsen et al. [1998] described a HT Abdominal Trauma Simulator (HATS), which was developed for open surgery from the front to remove a kidney that had been shattered as the result of blunt trauma. Berkley et al. [1999] described a simulator for in-wound suture training with real-time deformable tissues to increase the simulation fidelity.

Research on virtual bone surgery is still in its infancy. Most research on virtual bone surgery considered only deformation of muscles and cutting of soft tissues. Few have considered mechanical removal of bone material as shown in Figure 1.2, which is an integral part of bone surgery. Gibson et al. [1997] presented some early results of their effort to develop an arthroscopic knee surgery simulator. This simulator used a

volumetric approach to organ modeling. Computers were still too slow at that time to allow realistic deformation of a volumetric representation.

Agus et al. [2002] presented a haptic and visual implementation of a bone-cutting burr, which was developed as one component of a training system for temporal bone surgery, as shown in Figure 1.2 (a). They used a physics-based model to describe the burr-bone interaction, which included the haptic force evaluation, the bone erosion process, and some secondary effects such as the obscuring of the operational site due to the accumulation of bone dust and the resulting burring debris in the surgical penetration. Their implementation operates on a voxel discretization of patient-specific CT and MRI data and is efficient enough to provide real-time haptic and graphic rendering on a multi-processor PC platform. The ability of directly using patient specific data as input will help to accumulation a large number of training cases. The cutting tool is simplified as a sphere. However, volume modeling imposes strict requirements on computer memory and algorithm efficiency.

Morris et al. [2004] described a framework for training-oriented simulation of temporal bone surgery, as shown in Figure 1.2 (b). The system allows two users to observe and manipulate a common model in two different locations while performing a collaborative bone surgery and allows each user to experience the effects of forces generated by the other user's contact with the bone surface. This system also permits an instructor to remotely observe a trainee and to provide real-time feedback and demonstration. It uses hybrid data representation for graphic rendering and volumetric data representation for haptic feedback.

The Ohio Supercomputer Center and the Ohio State University Medical Center [Wiet et al., 2000; Bryan et al., 2001] developed a working prototype system for the simulation of temporal bone dissection, as shown in Figure 1.2 (c). It provides a customizable and more complete alternative to learning anatomy and surgical techniques, as opposed to printed media or physical dissections.

In the project IERAPSI (Integrated Environment for the Rehearsal and Planning of Surgical Intervention), John et al. [2001] and Jackson et al. [2002] focused on petrous bone surgery. The project brought together a consortium of European clinicians and technology providers working in this field. Three independent subsystems, pre-operation

planning, surgical simulation, and usage demonstration and training, were provided. Their work included direct volume rendering for visualization, and construction of a synthetic dissection workstation with an emphasis on training via a combination of identification and exposure of key structures, as well as integration of an intelligent tutor.

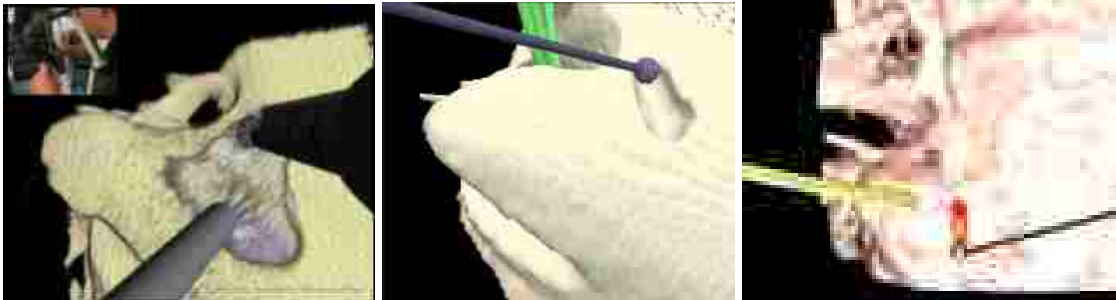


Figure 1.2. Some examples of temporal bone surgery simulators: (a) Agus et al [2002]; (b) Morris et al. [2004]; (c) Wiet et al. [2000] and Bryan et al. [2001]

Pflesser et al. [2002] and Petersik et al. [2002] presented a virtual petrous bone surgery system that allows realistic simulation of laterobasal surgical approaches. The system was based on a volumetric, high-resolution model of the temporal bone derived from CT scan data. Interactive volume cutting methods using a new multi-volume scheme were developed, allowing high-quality visualization of interactively generated cut surfaces.

Of all the simulators mentioned above, most of the research has focused on temporal bone surgery, and only a small portion of temporal bone was used in the simulation. The amount of data that needed processing was not huge, and tool-bone interaction was limited to burring/milling. In real orthopedic surgery, however, there are other machining operations like drilling, broaching, sawing, reaming, and milling, as illustrated in Figure 1.3. These operations, such as pin or screw insertion, are often

necessary prior to an orthopedic operation. One of the objectives of this research is to simulate such operations.

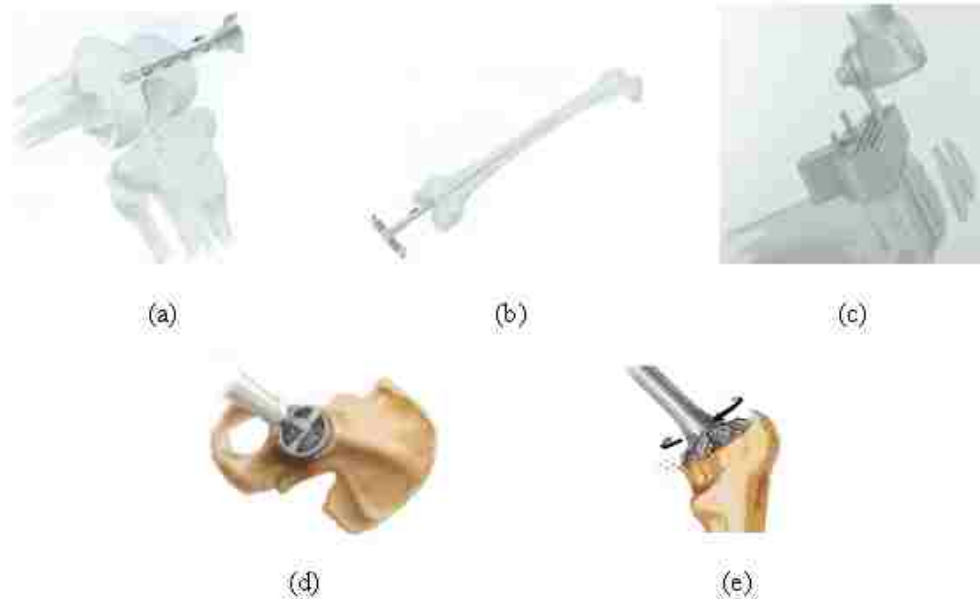


Figure 1.3. Typical machining operations in surgery of knee and hip bones: (a) drilling to enter the medullary canal; (b) broaching using an intramedullary rod; (c) sawing to cut off a piece of the distal femur; (d) reaming for acetabular preparation; and (e) milling for calcar planning

The simulation of material removal in bone surgery can be achieved in a manner similar to the simulation of a virtual sculpting process for creating 3D freeform objects from a stock. It should be noted, however, that bone surgery simulation deals with inhomogeneous materials, while virtual sculpting deals with homogeneous materials. The material removal can be simulated by continuously performing Boolean subtraction of the geometric model of the moving tool from the bone model. Galyean and Hughes [1991] introduced the concept of voxel-based sculpting as a method of creating freeform 3D shapes by interactively editing a model represented in a voxel raster. They developed a virtual sculpting system with a simple tool. Wang and Kaufman [1995] presented a

similar sculpting system featuring carving and sawing tools. In order to achieve real-time interaction, that system reduced the operations between the 3D tool and the 3D object to voxel-by-voxel operations. Bæntzen [1998] proposed octree-based volume sculpting and discussed its support of multi-resolution sculpting.

1.3. KEY TECHNOLOGIES IN BONE SURGERY SIMULATION

The schematic of a basic bone surgery simulation system is given in Figure 1.4. A personal computer based system is used to manipulate the interaction between the virtual bone and the virtual surgical tool and perform virtual bone surgery by “seeing” bone material removal through a graphic display device, “feeling” the force via a haptic device, and “hearing” the sound of tool-bone interaction.

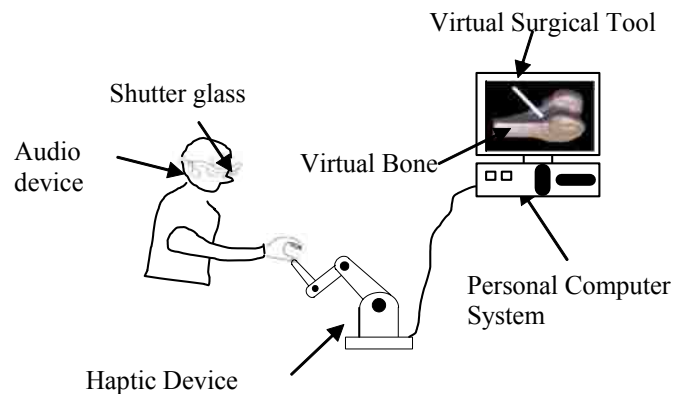


Figure 1.4. Schematic of a basic bone surgery simulation system

Generally, bone surgery simulation involves several issues: image acquisition and processing, geometric modeling, physical modeling, visualization, haptic interaction and sound rendering. The relationships between these key elements are illustrated in Figure 1.5. Image acquisition and processing usually precedes the simulation and is done off-line in order to save data processing time during on-line simulation.

A bone surgery simulation system consists of the following key elements as shown in Figure 1.5 [Leu et al., 2007]:

1. Input the CT or MRI data of the bones and construct a geometric model with attached properties such as material and density.
2. Develop physical models to represent the tool-bone interaction, of which the interface force is updated continuously, to simulate the deformation and cutting of bones, ligaments, etc.
3. Implement real-time graphic rendering of volumetric data to obtain realistic bone surgery visualization.
4. Provide force feedback to the user with haptic rendering.

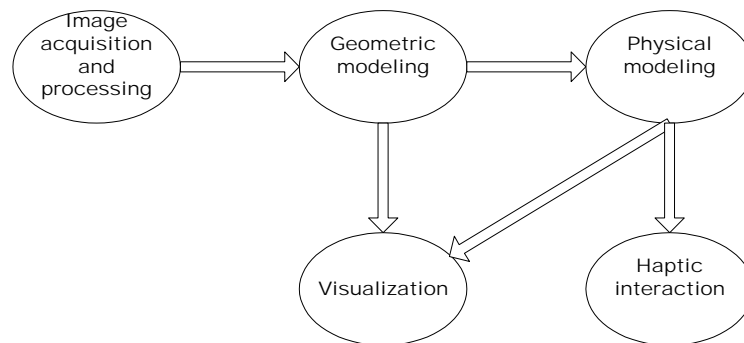


Figure 1.5. Key elements involved in bone surgery simulation

To provide a meaningful virtual bone surgery system with realistic force feedback and visual effects, several requirements that must be met:

1. The medical data obtained from image acquisition must be processed to minimize noise and irrelevant data [Jackson et al., 2002; Niu et al., 2005]. This data processing must be done before the bone surgery simulation.
2. In order to feed appropriate sensorial inputs to the human perceptual system, the simulation system must update the data at very different frequencies:

about 30 Hz for visual rendering and above 1k Hz for haptic response [Mark et al., 1996].

3. Interactive data modification is required for both visual and force feedback, so data modification calculation should involve only local data [Avila and Sobierajski, 1996; Astley and Hayward, 2000].
4. The amount of force computation time should be small for real-time haptic rendering [Avila and Sobierajski, 1996].

1.4. DISSERTATION LAYOUT

This dissertation presents the techniques and methods of modeling and rendering for the development of a virtual bone surgery system. Its organization is described below:

Section 2 presents an overview of the virtual bone surgery system. The objectives and challenges of this work are addressed first, followed by the system hardware and software configuration. The architecture of the simulation system is presented for design, research, and implementation purposes.

Section 3 describes image processing techniques, including feature extraction and region-based image segmentation for extracting both cortical bone and trabecular bone structures from CT scan data.

Section 4 describes the geometric modeling and the graphic rendering used to model the bone geometry from the CT scan data and to update the model during the simulated surgical operations.

Section 5 focuses on data management in an effort to reduce computation time for achieving real-time visualization based on the consideration of implementation complexity, memory storage, and computational overhead. Bounding volume and quadtree-based adaptive subdivision techniques are developed.

Section 6 covers surgical tool modeling and representation using implicit surfaces.

Section 7 discusses collision detection and force generation during tool-bone interaction. A multi-point collision detection algorithm and balance point based force calculation are developed for haptic rendering to make the surgery system more intuitive and interactive.

Section 8 presents the methods of sound acquisition, analysis, modeling, and rendering for the development of a virtual bone surgery system. Experiments conducted to record sound clips from drilling on different artificial bone materials are analyzed in both time domain and frequency domain to characterize the bone drilling sound. The real tool-bone interaction sound is obtained through spectral subtraction. The spectral modeling synthesis method is applied to get the sinusoidal model and residual model, which are then used for auditory rendering in the bone surgery simulation.

Section 9 covers the system implementation and integration and the results of developing a virtual bone surgery system.

Section 10 summarizes the dissertation and lists possible directions for future research.

2. VIRTUAL BONE SURGERY SYSTEM OVERVIEW

2.1. OBJECTIVES AND CHALLENGES

The major objective of this dissertation is to build a prototype virtual bone surgery system which can be used for training in orthopedic surgery, as well as for planning and rehearsal of bone surgery procedures. To develop of such a real-time interactive simulator, modeling and rendering techniques and methods are presented and developed in this dissertation

Designing and developing the VR surgery system described is a major undertaking. The dissertation faced a variety of challenging research issues related to the stringent requirements of real-time graphic, haptic and auditory rendering. The fundamental research issues include:

- Modeling the geometry of the bone generated from the CT scan data and modeling the dynamic change of this geometry due to material removal in virtual bone surgery.
- Representing, organizing, and manipulating the huge set of geometric and physical data in order to drastically reduce computations for real-time simulation and animation purpose.
- Modeling the tool-bone interface force considering both material removal and elastic deformation for common bone machining operations including drilling, reaming, milling, sawing, and broaching.
- Identifying the characteristics and features of bone drilling sound for different bone materials.
- Creating a sound model which allows fast real-time simulation, while also being sufficiently accurate to represent the important features of the drilling sound.
- Relating the generation of sound during the material removal operation to the bone-tool interface force, tool and bone vibrations, and other parameters such as bone and tool materials.

- Performing virtual reality rendering to simultaneously achieve the high-fidelity bone surgery visualization and a realistic feel and sound of the tool-bone interaction.
- Supplying a flexible, extensible and user friendly virtual bone surgery system.

2.2. HARDWARE AND SOFTWARE CONFIGURATION

The system's user interface is illustrated in Figure 2.1. The user can navigate in the virtual 3-D scene and do machining operations on the bone model. The bone model simulates a real bone structure: its boundary simulates the cortical bone structure, and its interior simulates the trabecular bone structure. With the system developed, the user can perform virtual bone surgery by simultaneously “seeing” bone material removal through a graphic display device, “feeling” the force via a haptic device, and “hearing” the sound of tool-bone interaction.



Figure 2.1. User interface of the virtual bone surgery system with the haptic device

Hardware and software configurations for system development and integration are shown below.

- Hardware:

Computer Desktop (single CPU at 2.8GHz, 512M RAM, RADEON 128 M graphic card, and Creative SB Live! sound card).

PHANToM Desktop (6 DOF/3 DOF haptic device).

- Software:

C/C++: main programming language;

MFC (Microsoft Foundation Classes): Windows-based programming library including Windows common dialogs, Windows common controls, Windows message processing, Document-View architecture, Event Handling, Mapping modes, Message map, Windows Modal Dialog and Common Controls, Graphic user interface related tools such as Scroll bar, combo box, radio button, and menus, keyboard accelerators, toolbars, status bars.

VTK (Visualization Tool Kit): an open source, object oriented software system from Kitware, Inc. for computer graphics, visualization, and image processing. It is a platform independent graphics engine with parallel rendering support. In this research, VTK is mainly used for graphics.

GHOST (General Haptics Open Software Toolkit): a C++ object oriented toolkit from SensAble Technologies, Inc. that represents the haptic environment as a hierarchical collection of geometric objects and spatial effects. It provides an abstraction that allows application developers to concentrate on the generation of haptic scenes, manipulation of the properties of the scene and objects instead of concerning with low-level force and device issues.

DirectSound API: audio component of MS-DirectX API, which provides Windows-based applications with high-performance low-level and real-time access to available multimedia hardware on a computer system in a device-independent manner. It enables hardware and software sound mixing and capture, and 3D positional effects.

2.3. VIRTUAL BONE SURGERY SIMULATION ARCHITECTURE

This project is aimed at designing and developing a realistic virtual bone surgery system by developing geometric and physical models and implementing them with

virtual reality technologies. The system consists of the following key components and functions: medical image processing and data management for simulation preparation, geometric modeling of bones and surgical tools' modeling and representation, physical modeling, and virtual reality rendering. The software architecture is shown in Figure 2.2. The developed system will work as follows. It constructs a geometric model of the bone from processed CT (Computed Tomography) image data. A virtual tool, such as a drill, is attached to the user's hand via a PHANTOM™ device (a product of SensAble Technologies), which is capable of providing 6D position and orientation data of the virtual tool and generating 3D force feedback to the user. The geometric bone model of is updated continuously during the virtual surgery process. A physical model is used to represent the tool-bone interaction, for which the interface force and sound generation are updated continuously. Virtual reality rendering is generated in real time to provide realistic visualization and force and sound feedback during the virtual surgery.

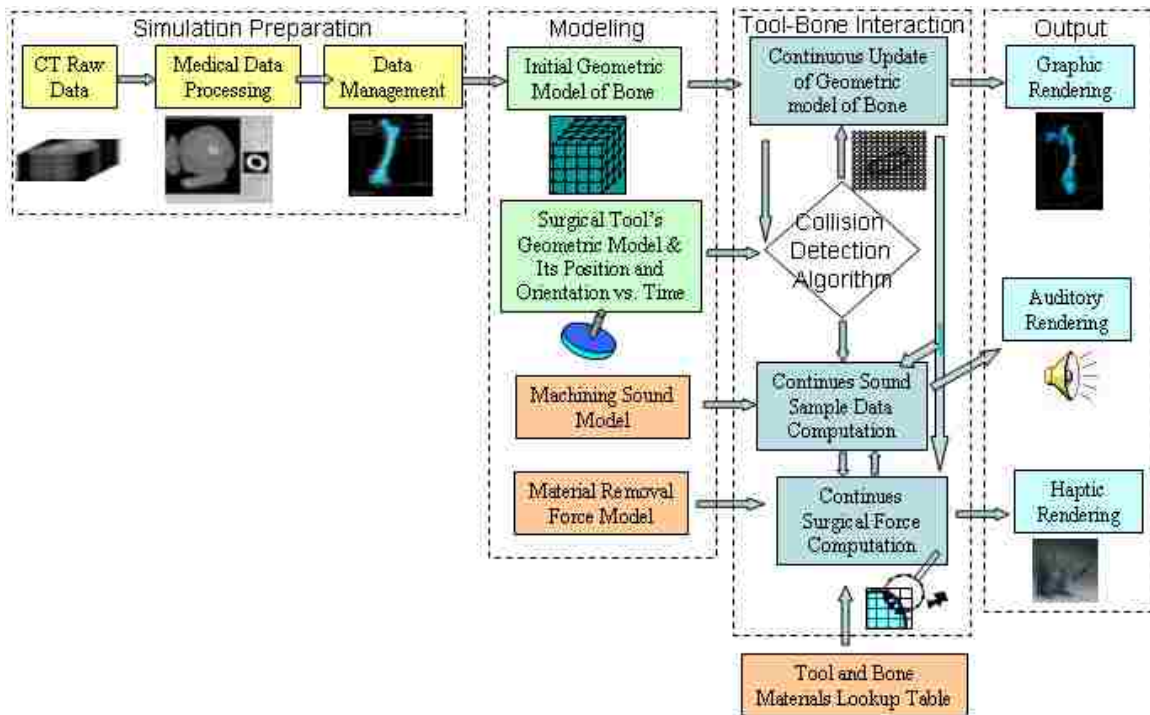


Figure 2.2. System architecture of the virtual bone surgery simulation system for this research

3. MEDICAL IMAGE PROCESSING AND SEGMENTATION

3.1. IMAGE PROCEDURES

Computer imaging techniques have become important diagnostic tools in the practice of modern medicine. Today, advanced medical scanners can provide high-quality, highly detailed images for surgeons to evaluate prior to performing the actual physical procedures. Medical data obtained from imaging techniques typically represent the values of some properties at various locations [Kaufman et al., 1993]. The most commonly used medical imaging techniques include CT (Computed Tomography), MRI (Magnetic Resonance Imaging), SPECT (Single-Photon Emission Computed Tomography), and PET (Positron Emission Tomography), as shown in Figure 3.1. These techniques use a data acquisition process to capture information about a patient's internal anatomy. This information is in the form of slice-plane images, similar to conventional photographic X-rays [Schroeder et al., 2002].

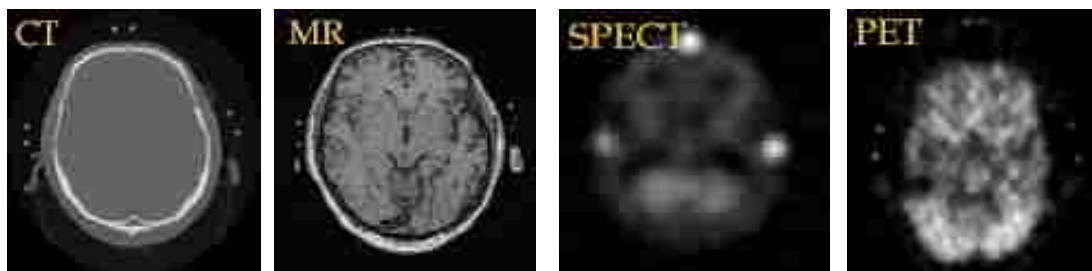


Figure 3.1. The most commonly used medical imaging techniques (<http://www.cs.uu.nl/docs/vakken/imtw/>)

CT and MRI are most commonly employed to obtain medical images. CT provides high spatial resolution bone images, while MRI provides better images for soft tissues. CT scan data are used for most bone surgery simulators in the research because they show better contrast between bones and soft tissues. For reporting and displaying reconstructed CT values, Hounsfield Unit (HU) is usually used. This unit describes the

amount of x-ray attenuation of each volume element in the three-dimensional image. There are good correlations between CT scan data and bone's material properties such as density and mechanical strength [Bentzen et al., 1987], so HU value is usually used for each data point to represent bone density.

CT scanners are usually calibrated only by two points of -1000 for the attenuation of dry air and 0 for that of pure water at 25⁰C at the effective scanning energy used. Table 3.1 [Gladilin, 2003] shows an example of the approximate range of HU values corresponding to the different materials and tissue types.

Table 3.1. HU values and different materials and tissue types

Tissue	HU Values
Air	-1000
Lungs	-1000 to -400
Fat	-100 to -50
Water	0
Brain	0 to 100
Muscle	10 to 600
Soft Tissue	-100 to 300
Bone	>500

Many researchers present a linear relationship between bone mineral density (BMD) and CT HU values:

$$\text{BMD (mg/cc)} = (H-a) / k \quad (3.1)$$

where H is the CT HU value and slope k and intercept a can be calculated with the mean CT values of standard and their BMD equivalent concentration. For example, in a well-

calibrated scanner at ideal conditions, -1000 HU is the value given to air (1kg/m^3) and 0 HU given to water (1000 kg/m^3), so the linear relationship between BMD and CT values can be found.

The process of constructing a VR environment from imaging data is a major challenge. The process can be divided into three stages: 1) spatial co-registration of data from multiple modalities; 2) identification of tissue types (segmentation); 3) definition of tissue boundaries for the VR environment [Jackson et al., 2002].

3.2. IMAGE PROCESSING AND SEGMENTATION

Noise and other artifacts are inherent in all methods of data acquisition. Due to noise in many signals and lots of irrelevant information in the medical data, image processing is necessary. Filtering and smoothing techniques, e.g., Gaussian filters and median filters, are usually used to reduce noise on images [Schroeder et al., 2002]. After Gaussian filter and edge detection are used to filter/threshold the image data slice by slice, the noise is removed to represent the specific regions of a bone. Figure 3.2 shows an example of image data before and after the processing. As shown, only a small portion of the slice is used to construct the bone's volumetric model and later used for collision detection, force generation, and model update in the bone surgery simulation. A large amount of irrelevant data is not used in constructing the bone model.

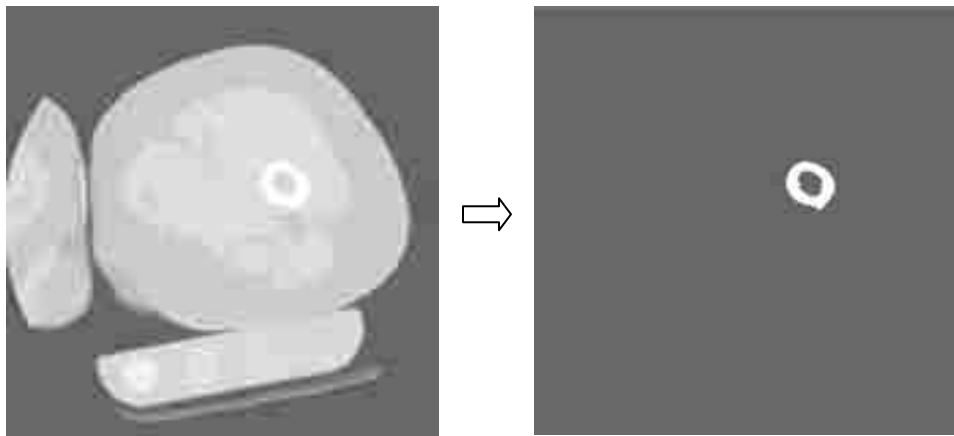


Figure 3.2. Example of image data before and after processing

Since information gained from various images acquired through a medical imaging procedure is usually complementary, proper integration of useful data obtained from the separate images is desired. Image reiteration is the process of determining the spatial transform that maps points from one image to homologous points on the second image [Luis et al., 2003]. These images' formats may or may not have the same format. The most common registration methods could be found from the survey of medical image registration by Maintz and Viergever [1998].

It is also necessary to identify which type of tissue is present in the data space and to identify the precise location of edges between different tissue types. Image segmentation is the process of identifying the distribution of different tissue types within the data set. Bones can be extracted by manual or partially automated segmentation methods. Threshold segmentation is usually used to distinguish pixels or voxels within an image by their gray-scale values. An upper or lower threshold can be defined, separating the image into the structure of interest and background. This method works well for bone segmentation from CT scans since bone tissue attenuates significantly more during image acquisition and is, therefore, represented by much higher values on the Housfield scale compared to soft tissues. Whereas thresholding focuses on the difference in pixel intensities, segmentation looks for regions of pixels or voxels with similar intensities [Ritter et al., 2004].

After the noise reduction, the region of interest (ROI) is extracted by using a bounding box, which bounds the bone and a portion of the soft tissue. An image segmentation technique is used to divide the data into bony structures (femur, tibia, etc.), cartilage, tissues, and ligaments. Segmentation methods are usually divided into two kinds: region-based and edge-based [Kovacevic et al., 1999]. Region-based methods search for connected regions of pixels/voxels with some similar features, such as brightness, or texture pattern. After dividing the medical image into regions in some manner, similarity among pixels is checked for each region, then neighboring regions with similar features are merged into a bigger region and a region with different features within it is split into smaller regions. These steps are repeated until there is no more splitting or merging. A main problem in this approach is how to determine an object's exact borders because regions are not necessarily split on natural object borders. Edge-

based algorithms search for pixels with high gradient values, which are usually edge pixels, and then try to connect them to form a curve which represents a boundary of the object. Connecting high gradient pixels is a different problem because in real images they might not be neighbors.

A region-growing algorithm is implemented for the medical image segmentation, as shown in Figure 3.3.

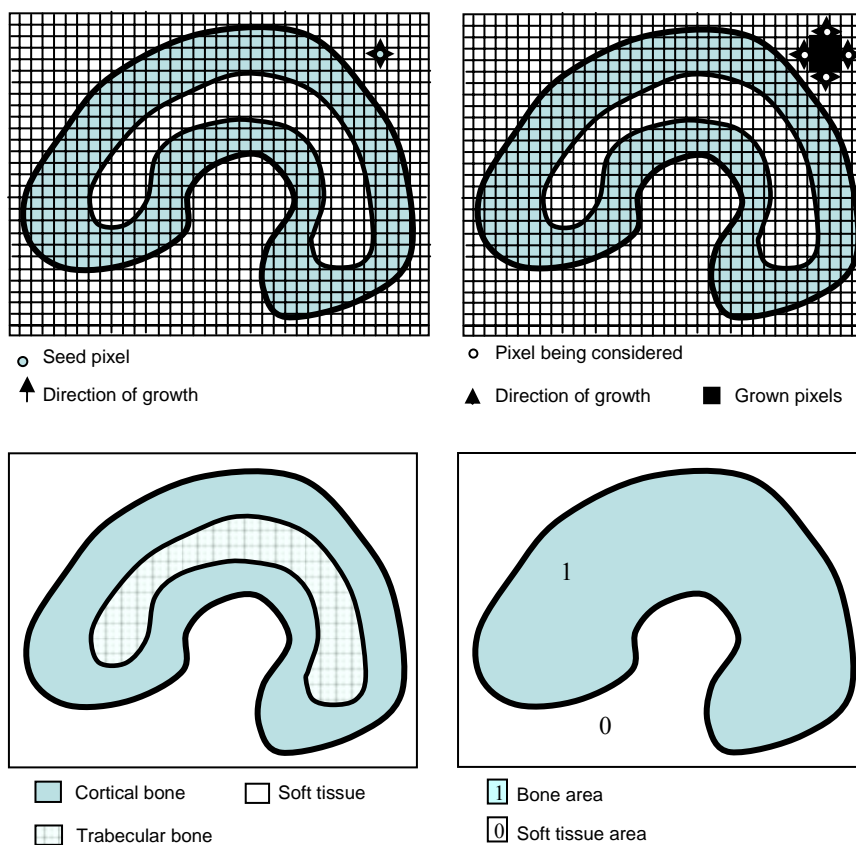


Figure 3.3. Region growing image segmentation algorithm

The main idea behind this algorithm is to identify various regions in an image that have similar features [Kovacevic et al., 1999]. Briefly, the algorithm consists of the following steps:

1. After selecting a seed pixel within the image and comparing it to neighboring pixels, a region is grown from the seed pixel by adding in similar neighboring pixels.
2. Each pixel in the image receives a label from the region growing process with only pixels belonging to the same region having the same label.
3. When the growth of one region stops, another seed pixel which does not yet belong to any identified region is chosen and the region-growing process starts again.
4. The whole procedure is finished when all the pixels have been examined. During implementation, a first-in-first-out queue, rather than a recursive stack, is used as the data structure to avoid memory/stack overflow.

4. GEOMETRIC MODELING AND GRAPHIC RENDERING

4.1. GEOMETRIC MODELING

The sequence of 2D data slices obtained by CT or MRI can be represented as a 3D discrete regular grid of voxels (volume elements), as shown in Figure 4.1. For surgical simulation, voxel-based modeling has a number of advantages over the use of surface polygons or solid primitives. First, voxel-based representation is natural for 3D digital images obtained by medical scanning techniques such as MRI or CT. Second, since no surface extraction or data reformatting is required, errors introduced by fitting surfaces or geometric primitives to the scanned images can be avoided. Third, volumetric objects can incorporate detailed information about the internal anatomical or physiological structure of organs and tissues. This information is particularly important for realistic modeling and visualization of complex tissues [Gibson et al., 1997].

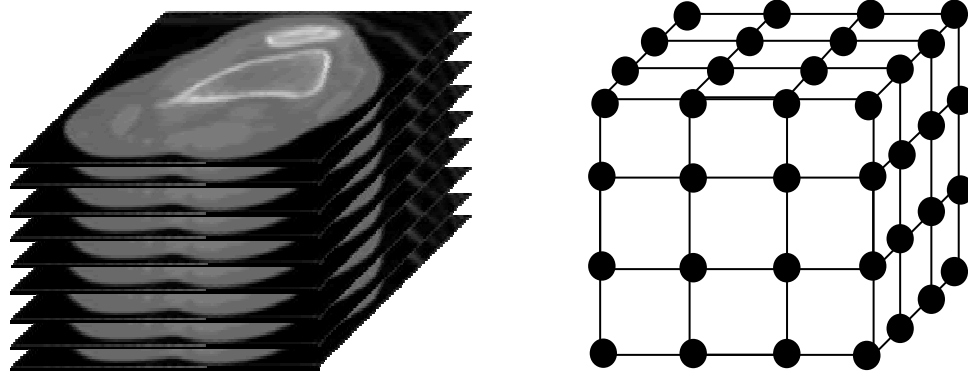


Figure 4.1. A volume seen as a stack of images and a volume seen as a 3d lattice of voxels

In volume representation the basic elements are voxels [Bærentzen, 2001]. Just as a pixel is a small rectangle, a voxel can be viewed as a small block. A voxel can be represented by the coordinates of its center point and the three orthogonal dimensions

plus some attributes. If the voxels have fixed dimensions, then they can be represented by the vertices of a 3-D lattice, which are characterized by their positions and the associated values of their attributes. For example, voxels can be expressed as an array $(x, y, z, v_1, v_2, \dots, v_n)$ where (x, y, z) represents the position of each voxel and v_i represents a property. These properties can be physical properties such as density, material classification, stiffness, and viscosity, or display properties such as color and shading.

In general, the samples may be taken at random locations. Depending on how the samples are connected to form a grid structure, there are two classes of volumetric data: structured and unstructured. Structured data has a logical organization of the samples into a three-dimensional array and a mapping of each sample into the physical domain. Unstructured data is not based upon a logical organization of arrays, but instead upon a group of cells of certain shapes such as tetrahedra, hexahedra, or prisms.

An interpolation function is used to produce a continuous scalar field for each property. This is critical for producing smooth volume rendering and haptics rendering [Avila and Sobierajski, 1996]. In order to meet the system requirements, it is often desirable to pre-compute and store the contents of each voxel to avoid changing every voxel during the surgical operation simulation. By storing the volumetric data in a space-efficient, hierarchical structure, the storage requirements can be reduced.

The following definitions are used for the purpose of later discussion of image data management and other techniques and methods.

Voxel: A voxel is expressed as an array $(x, y, z, v_1, v_2, \dots, v_n)$, where (x, y, z) represents the position of each voxel and v_i represents a property such as color, density, or material strength. Voxels are used to represent image data obtained from CT scans.

Cell (Volumetric Cell): Eight voxels with their attributes, e.g., HU values for density, are stored at the corners of a Volumetric Cell. Each cell has six faces, twelve edges, and eight corners.

Sub-volume: A sub-volume is a block which has a set of adjacent cells. The smallest sub-volume is one cell.

Whole-volume: The whole-volume is the entire model of the concerned object. The whole-volume consists of all the sub-volumes, and each sub-volume may have different dimensions.

Voxel modeling has many advantages such as unambiguity, heterogeneity, simplicity in Boolean operations, excellent morphological dexterity, and excellent local editability. However, algorithms based on elementary voxel modeling techniques with uniform voxel sizes often pose large memory storage and real-time performance issues [Borro et al., 2004]. Chapter 5 will describe these issues.

4.2. GRAPHIC RENDERING

Volume visualization is a technique used to display the information inside volumetric data using interactive graphics and imaging. Basically, the methods of graphic rendering of three-dimensional data (volumetric data) fall into two categories: (1) Surface Rendering or indirect rendering and (2) Volume Rendering or direct rendering. In deciding which method is suitable for the bone surgery system, the following considerations are important: (1) real-time rendering and (2) surface quality in graphic display. Surface rendering extracts polygons from volumetric data and renders the surface interactively. Volume rendering cannot give interactive real-time performance based on current graphics hardware and software unless very coarse approximation is made.

4.2.1. Surface Rendering. Marching Cube [Lorensen and Cline, 1987] is the most popular algorithm used in surface rendering. The marching cube algorithm traverses all boundary cells of the entire volume and determines the triangulation within each cell based on the values of cell vertices. This method first partitions a volume data into cubes with each cube consists of eight voxels. Then it determines the surface configuration of each cube according to 15 possible configurations, as shown in Figure 4.2. The marching cube leads to satisfactory results for small or medium datasets. However, for simulation in the medical field, data sets are usually quite large and can restrict the real-time performance of interactive manipulation. The use of octrees for faster isosurface generation proposed by Wilhelms and van Gelder [1992] is an improved algorithm for extracting surfaces from volumetric data. This algorithm stores min/max voxel values at each octree node, then traverses octree nodes that may contain an isosurface to obtain the triangles that form the surface. Other researchers [Shekhar et al., 1996; Sutton and Hansen, 1999; Velasco and Torres, 2001] have also presented improved octree-based marching cube algorithms and their applications. These methods use some techniques to

save data storing space and improve computational performance, but they do not support multi-resolution isosurface extraction.

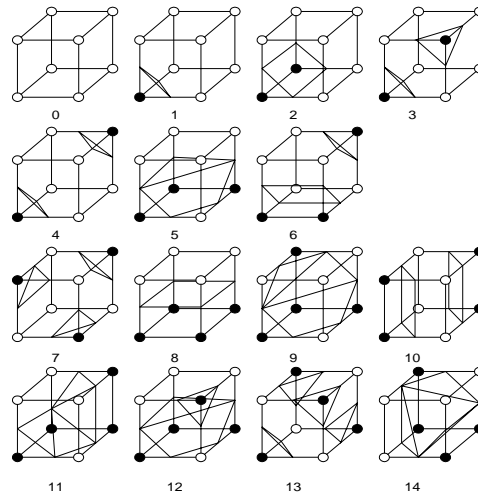


Figure 4.2. The marching cube algorithm for surface rendering of voxel data

Adaptive resolution surface rendering is the method most commonly used in bone surgery simulation. Some researchers [Westermann et al., 1999; Boada and Navazo, 2001] have presented ideas on this rendering method. The rendering algorithms are based on an extended marching cube algorithm for octree data, as follows:

1. The focus of interest is located (i.e. the current surgical tool location and its neighborhood);
2. The region of interest is rendered in high resolution, meaning that the cells in the region are subdivided into sub-cells and the surface is extracted at the sub-cell level using the marching cube algorithm;
3. The rest regions are rendered in lower resolution. The cells in these regions are merged to form coarser cells.

A trade-off exists between surface quality and interactivity. Although octree can address this problem to some extent, interactivity is still difficult to achieve for a large

data set. In order to improve performance, the initial resolution (usually not very fine) for the surface rendering needs to be specified. Dynamic resolution can be used depending on how the surgical tool interacts with the bone material. Parallel computing can be used to increase the computation and, hence, the resolution.

4.2.2. Volume Rendering. In this method the volume data are directly rendered, which means that the images are generated through the transformation, shading, and projection of 3D voxels into 2D pixels. Volume rendering demands greater computational processing, but produces images with greater versatility. Since all the voxels located in the line of view are used in image generation, this method allows the visualization of parts inside the surface. Although real-time rendering can hardly be achieved, this method is a good choice for some applications with special visualization requirements. Volume rendering will become more attractive in the future as computers become faster, cheaper, and have larger memory.

The most popular algorithm for volume rendering is *Ray-Casting* [Levoy, 1988 and 1990]. Traditionally, the ray-casting algorithm spans the projection plane and traces the rays into the scene. Usually, parallel rays that are orthogonal to the projection plane are cast. These rays are traced from the observer position to the volume data. For each ray, sample points are calculated considering a fixed step on the path. The algorithm can calculate and accumulate both color and opacity values along the ray to obtain the pixel color. Besides ray-casting, other popular algorithms in the volume rendering approach include splatting [Westover, 1990], shear-warp [Lacroute and Levoy, 1994], and 3D texture-mapping [Cabral et al., 1994]. Meißner et al. [2000] conducted an extensive survey on volume rendering algorithms.

Most bone surgery simulation systems do not use volume rendering because of the interactivity restriction, the need for expensive, dedicated graphics hardware, and the need for large amounts of computation time and storage space. However, the merits of volume rendering, along with the continuing decrease of computation costs, may compel researchers to use this method in the future. In this dissertation, the marching cube algorithm is used as surface rendering technique for graphic rendering.

5. LARGE MEDICAL DATA MANAGEMENT

5.1. INTRODUCTION

The data set for surgery simulation is usually very large. For example, for a medium resolution of 512^3 , two bytes per voxel, the volume buffer must have 256M bytes [Kaufman et al., 1993]. The organization and manipulation such huge amount of data is a challenging problem.

Zhu et al. [1998] used a finite element method (FEM) of analysis in their study of muscle deformation. A muscle was modeled with 8-node, 3D brick elements equivalent to the voxel structure. The simulation was achieved by solving a sparse linear system of equations which governs the model's behavior. Gibson et al. [1997] developed a linked volume model to represent the volumetric data. The links were stretched, contracted, or sheared during object deformation and were deleted or created when objects were cut or joined. Compared to the FEM method, the linked volume approach can be used to create models with high geometric complexity and can achieve interactivity through the use of low-cost mathematical modeling.

Bæentzen [1998] proposed an octree-based volume sculpting method for quickly separating homogeneously empty regions outside the object of interest. As shown in Figure 5.1, an octree structure was chosen to organize the huge set of volumetric data and to improve the efficiency of data storage. A volume was subdivided until the leaf level of a prescribed size had been reached. This technique can significantly reduce the memory requirement and speed up the graphic rendering and the modeling task. Basically, octrees are a hierarchical variant of spatial-occupancy enumeration that can be used to address the demanding storage requirements in volume modeling [Foley et al., 1996].

In bone surgery simulation, operation tools such as drills, mills, and broaches remove voxels occupied by the cutting tool's volume during the course of operation. For static data structure, e.g. 3D arrays, voxels can only be removed in a defined size. That is, the cells representing the interaction between the cutting tool and the bone are constant in size, so the resolution is static. Due to this limitation, voxel removal can only be done on a rough level. Octree modeling can provide a flexible data structure for dynamically performing material removal simulation. High resolution can be achieved in the region of

interest, which is usually the current surgical tool location and its neighborhood. The octree nodes representing cells in the region of interest are subdivided to generate child nodes representing sub-cells. The material removal operation is then conducted at the child node level. The subdivision process can be repeated until the desired resolution is reached. A criterion to end the subdivision can be set to control the resolution automatically. The criterion could be that the smallest linear dimension of the voxel is equal to the radius of the drill multiplied by a sizing factor.

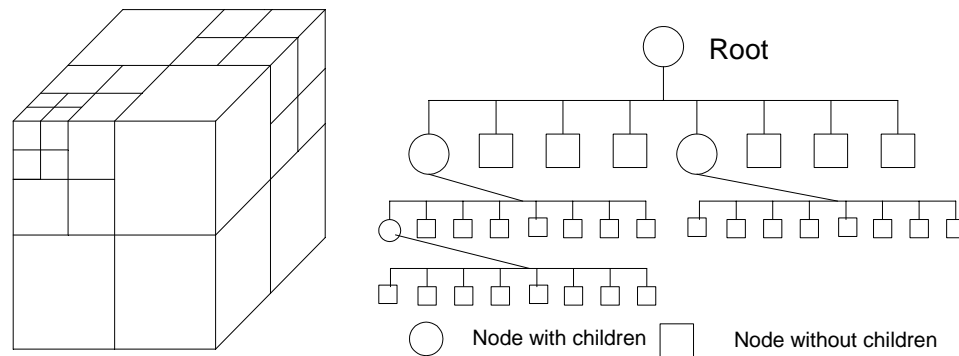


Figure 5.1. Octree representation

High visualization quality and real-time display are essential to develop a virtual bone surgery system. The volumetric data set from CT scans is commonly $128 \times 128 \times 128$, but can be as large as $512 \times 512 \times 512$. Challenges in processing the large data set include:

1. Memory. For the CT scans used in this study, the size of each slice is 512×512 and there are more than 256 slices. If all the data is input as source data, the set will have more than 67×10^6 data points.
2. Preprocessing time. Because there are many input files and the data set is very large, the preprocessing of data before the surgery simulation may take a long time.

3. Real-time performance. During the surgery simulation, the virtual tool will interact with the bone model when performing material removal. The data calculation and update, graphics rendering, etc., will use a large amount of computational resources. Too many active voxels will influence real-time performance;
4. Resolution. The voxel size is defined as $0.489\text{mm} \times 0.489\text{mm} \times 1.5\text{mm}$ in the original CT scan data. This resolution may not be fine enough, but a finer resolution will demand the input of an even larger data set.

Because a huge set of volumetric data is involved in the simulation, it is important to organize and manage these data carefully. Considering the volumetric modeling, graphics rendering, material removal process, complexity of implementation, storage and computational overhead, etc., this dissertation handles the very large data set by reducing irrelevant input data and developing an efficient data structure. Two techniques, bounding volume and quadtree-based adaptive subdivision, are developed for these purposes.

5.2. DATA MANAGEMENT METHODS

5.2.1. Bounding Volume. A volume buffer or cuberille [Foley et al., 1996], which has a structured data type ($N_x \times N_y \times N_z$), is often used to store volumetric data in computer memory. After preprocessing the CT scan data, the boundary of the bony area of each slice and the new height for the Z direction (starting from slice number l_s , ending with slice number l_e) can be obtained, then a volumetric model can be constructed using of a bounding volume to get rid of some irrelevant data. Bounding volume is one of the important techniques used in Computer Graphics. There are several different bounding volume types, such as Oriented Bounding Box (OBB), Bounding Sphere (BS), Bounding Ellipsoid (BE), and Axis Aligned Bounding Box (AABB) [Foley et al., 1996]. AABB was chosen for this study because of the CT setup and the fact that AABB is the easiest one to implement. The dimensions of the bounding volume are determined after finding the maximum and minimum values from the data processing for the bone model construction.

An example of bounding volume selection is shown in Figure 5.2. For each of the i slices of the CT scans (i could be any continuous slices from l_s to l_e), the left-most and the right-most bony data point are marked as X_i^{MIN} and X_i^{MAX} for X direction, and the upper and the lower ones are marked as Y_i^{MIN} and Y_i^{MAX} for Y direction. After projecting the bone boundary onto the X-Y plane, a new projected bone boundary can be obtained. The new bounding volume of this block is from $\min(X_i^{MIN})$ to $\max(X_i^{MAX})$ in X direction, from $\min(Y_i^{MIN})$ to $\max(Y_i^{MAX})$ in Y direction, and i in Z direction. Note that $\max(\cdot)$ and $\min(\cdot)$ are two functions which can obtain the maximal and minimal value from the given input data.

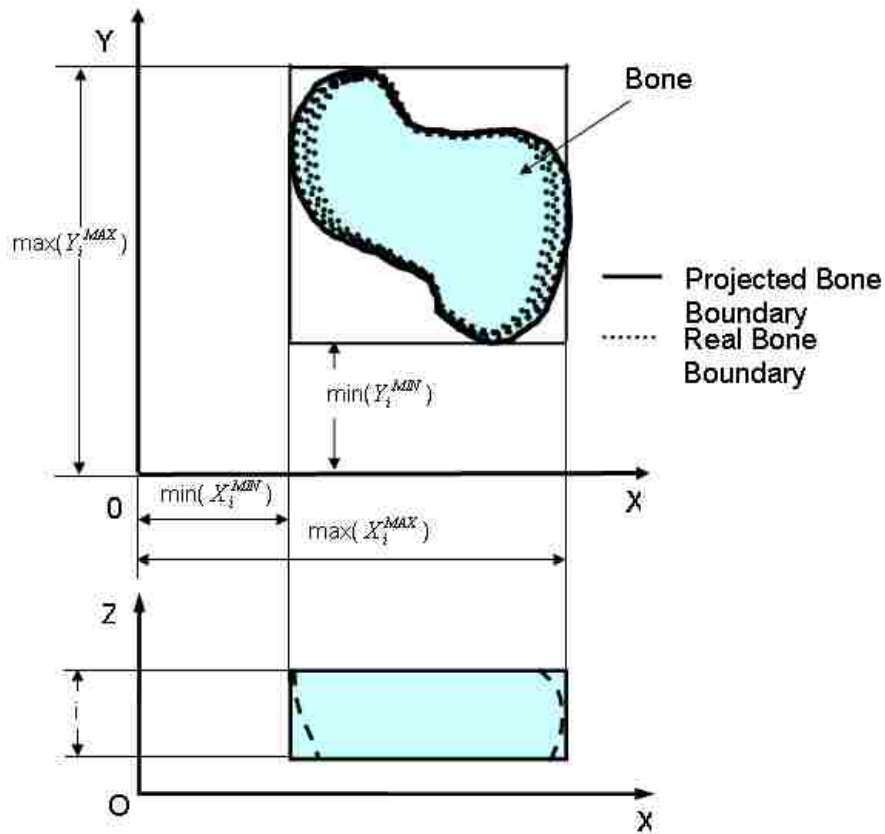


Figure 5.2. An example of bounding volume selection

Figure 5.3 shows the bounding volumes of a bone. If i is chosen from l_s to l_e , it is easy to determine only one big bounding volume called the object bounding box (OBB), which contains all the bony area to construct the volumetric model of a femur, as shown in Figure 5.3 (a).

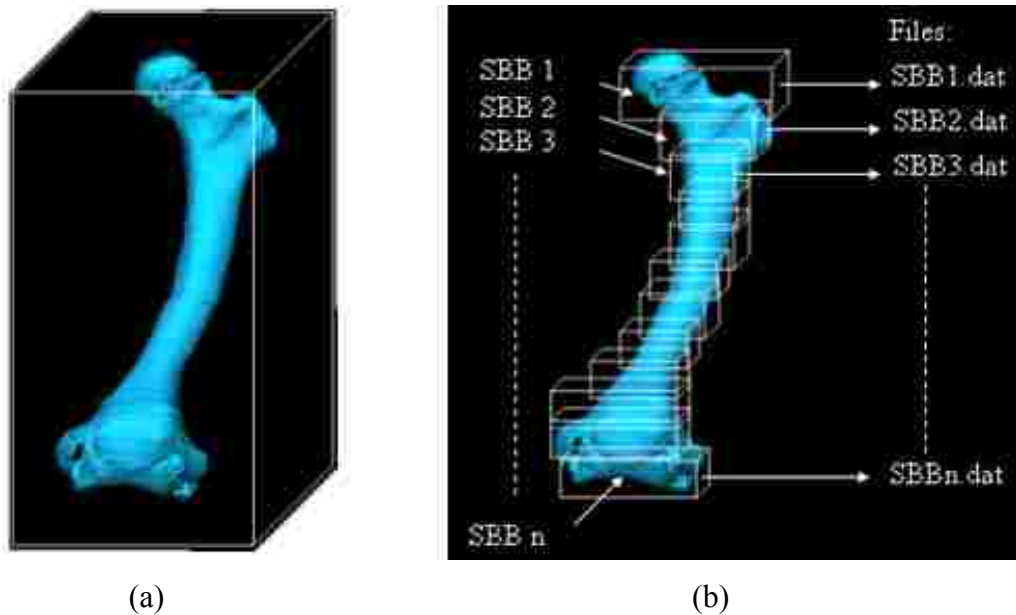


Figure 5.3. Bounding volumes of a bone: (a) one single bounding volume, and (b) multiple sub-volumes

The key to determining the bounding volume size is finding a tight bounding box for the bone's image data. The bounding box may still contain much irrelevant data because the human skeletal shape is irregular. To update and render the data in a large single bounding volume is still time-consuming, so it is difficult to perform real-time simulation. Hence, a refined bounding volume technique is developed by choosing multiple slices of volume as shown in Figure 5.3 (b). After image processing, the whole bone is divided into many sub-volumes, which have certain layers in Z direction and different dimensions in X and Y directions. These sub-volumes (from 1 to n), called the

section bounding box (SBB), have relatively tight bounding boxes around the objects. To determine the X and Y dimensions of the a sub-volume, the selection method mentioned above is used to check the extreme X and Y values slice by slice, then find the maximum and minimum X and Y values among all the obtained values and use them to make each bounding sub-volume as tight as possible. An efficient algorithm is developed and implemented to determine the smallest size of the whole data set and the height of each sub-volume.

In order to save the simulation's data loading time, the image processing and the sub-volume creation processes are conducted before the simulation starts. Two different kinds of files are used for data control and data storage: one Meta file and several root sub-volume data files (from 1 to n). The Meta file is used to control the sub-volume data files. It includes information such as the file directory, stored sub-volume files, number of root sub-volume files, start slice number, slices for each root sub-volume, and subdivision times for each sub-volume data. The data of each sub-volume, from sub-volume 1 to sub-volume n , are saved into the files with names SBB1.dat, SBB2.dat, ..., SBBn.dat, as shown in Figure 5.3 (b), and then reloaded as new input data before the simulation. These n sub-volume data files have similar file structures but different numbers of medical data points. The sub-volumes' file structure is as follows:

```
[File Structure: Sub-volume files]
DATASET SUB-VOLUME DATA
DIMENSIONS  $n_x n_y n_z$ 
SPACING  $s_x s_y s_z$ 
ORIGIN  $x_o y_o z_o$ 
DATA_NUMBER  $d_n$ 
BOUNDARY POSITION
0  $x_{01} y_{01} x_{02} y_{02}$ 
1  $x_{11} y_{11} x_{12} y_{12}$ 
.....
.....
 $n_y x_{ny1} y_{ny1} x_{ny2} y_{ny2}$ 
MEDICAL DATA
.....
(HU values for the total  $d_n$  data points)
.....
```

5.2.2. Quadtree-Based Adaptive Subdivision. The data need to be updated at each time step for the simulation of material removal, such as drilling in bone surgery. When the data are changed during the simulation process, the graphics need to be re-rendered for display. However, editing a whole object volume for display would be impractical. It is desirable to display a substantially smaller volume in order to achieve real-time performance. Instead of dealing with the entire volume, only the sub-volumes of interest are dealt at each time step. Thus, the data set being processed at each time step could be drastically reduced and the system's real-time performance could be significantly improved. Figure 5.4 shows the division of a complete model into uniform sub-volumes, with each sub-volume consisting of many voxels. It also shows a sub-volume becoming an active sub-volume when the tool enters its region. There is more than one active sub-volume when the tool is at the boundary of two or more sub-volumes, as illustrated on the right side of Figure 5.4. Only active sub-volumes are updated in the display at each time instant during the simulation. This saves more time than updating the complete model at every time instant. A 3D array is usually used to handle the storage for a uniform division.

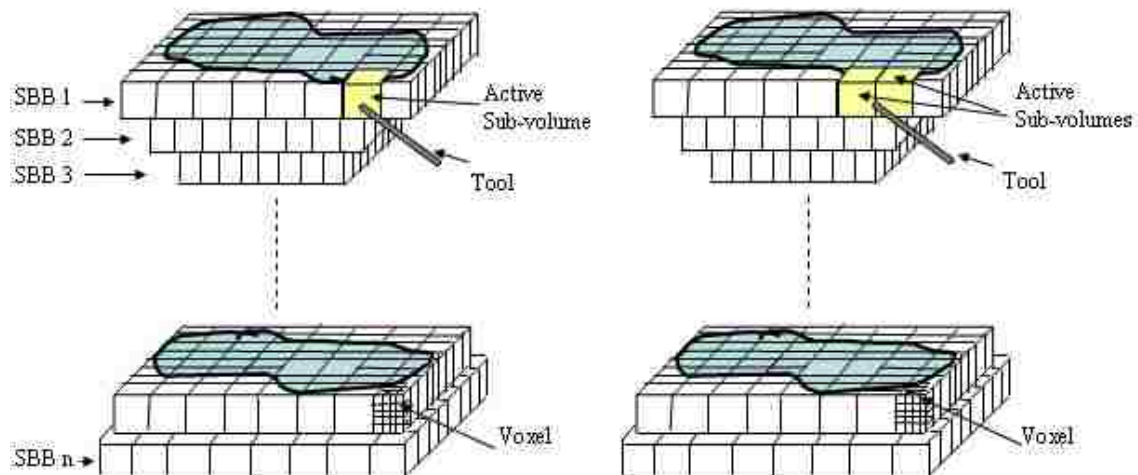


Figure 5.4. Uniform division of a model into sub-volumes

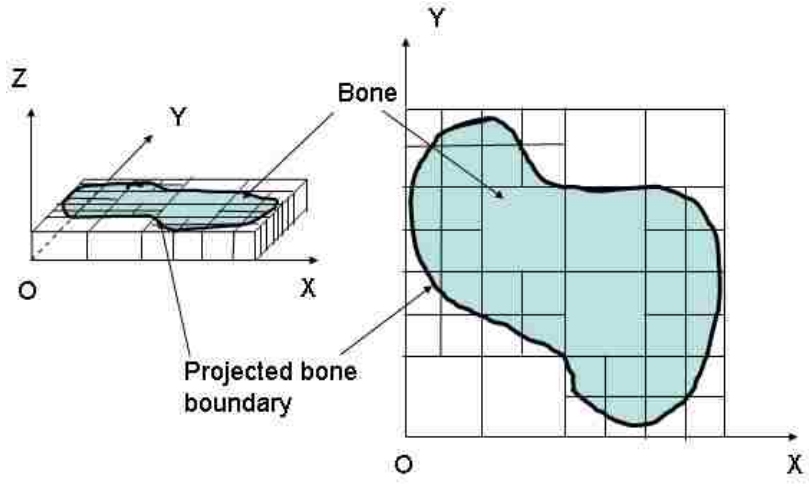
The division of a complete model into uniform sub-volumes, as shown in Figure 5.4, is called uniform division or spatial occupancy enumeration. The key advantage of this technique is that it allows data traversal in the sub-volume grid. Its disadvantages include lower computational efficiency and larger memory requirement (especially for objects with odd shapes), and non-adaptation (e.g., empty sub-volumes are hard to delete).

Although the whole bone model has been divided into sub-volumes, the data set is still too large to achieve real-time simulation. This dissertation uses adaptive partitioning to deal with this problem. Quadtree Subdivision is chosen as the adaptive partitioning method because the whole bone model has already been divided into sub-volumes with small dimensions in z direction.

Quadtree is a 2D Octree frequently used to save memory space in computer graphics. The fundamental idea behind both quadtree and octree subdivision techniques is the divide-and-conquer power of binary subdivision [Foley et al., 1996]. A quadtree is derived by successively dividing both x and y dimensions to form quadrants. Each quadrant of the sub-volumes may be full, partially full, or empty, depending on whether the sub-volume of consideration intersects the area of concern. A partially full quadrant is recursively subdivided into sub-quadrants. The subdivision continues until all quadrants are homogeneous or until a predetermined cut-off resolution is reached. The successive subdivisions can be represented as a tree with the partially full quadrants as the internal nodes and the full or empty quadrants as the leaves. The division of an SBB three times with the corresponding tree structure is illustrated in Figure 5.5. As shown in the figure, if the subdivision number is 1, all the quadrants are partially full. After it is subdivided twice, four “empty” sub-quadrants and three “full” sub-quadrants appear. During the third subdivision, only the “partially full” quadrants resulted from the second subdivision need to be considered. The “partially full” quadrants are usually on the boundary of the bone model and all “full” quadrants are usually inside. The data for “empty” quadrants can be deleted during the subdivision process to save further memory.

Compared to the uniform division technique, Quadtree Subdivision can reduce the amount of memory required to represent the bone model, improve computer execution times for querying and processing data, and also reduce computational overhead [Frisken

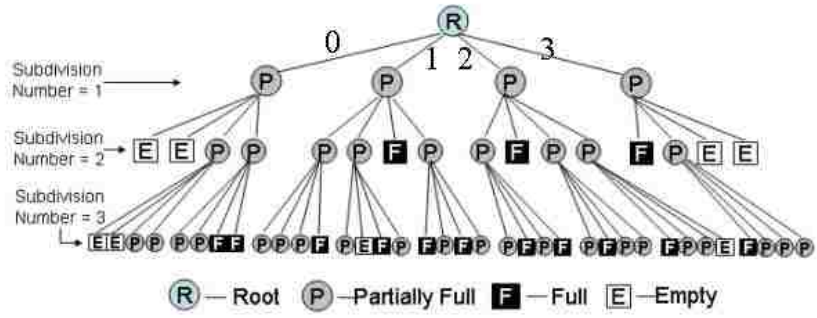
and Perry, 2002]. Other advantages of the tree structure include scalability, flexibility, and ease of implementation.



(a)

2	3
0	1

Quadrant Numbering



(b)

Figure 5.5. An example of quadtree subdivision

5.2.3. Data Structure. A data structure is developed for the proposed techniques of image processing, bounding volumes, and quadtree subdivision. The processed image data with bounding sub-volumes generated is treated as a new set of input data for the

simulation, as shown in Figure 5.6. Each of the sub-volumes (SBBs) is an object of a sub-volume class and its data is stored in a LIST structure. Each SBB (from 1 to n) can be divided into smaller sub-volumes according to the quadtree subdivision rule. These smaller sub-volumes are still objects of the sub-volume class. To construct the presented data structure, it is necessary to follow these steps:

[Algorithm: Data structure construction]

1. Read Meta file and get controlling information, such as file directory, number of root sub-volume files, etc;
 2. Create a new CSubcolume list l for these n sub-volumes;
 3. For loop from 1 to n
 - Read sub-volume file and get info for each root sub-volume;
 - Create a new CSubvolume object s and Set new Origin, Dimensions, etc;
 - Deep copy medical data to s;
 - s->subdivide();
 - put s to the list l;
- end for loop

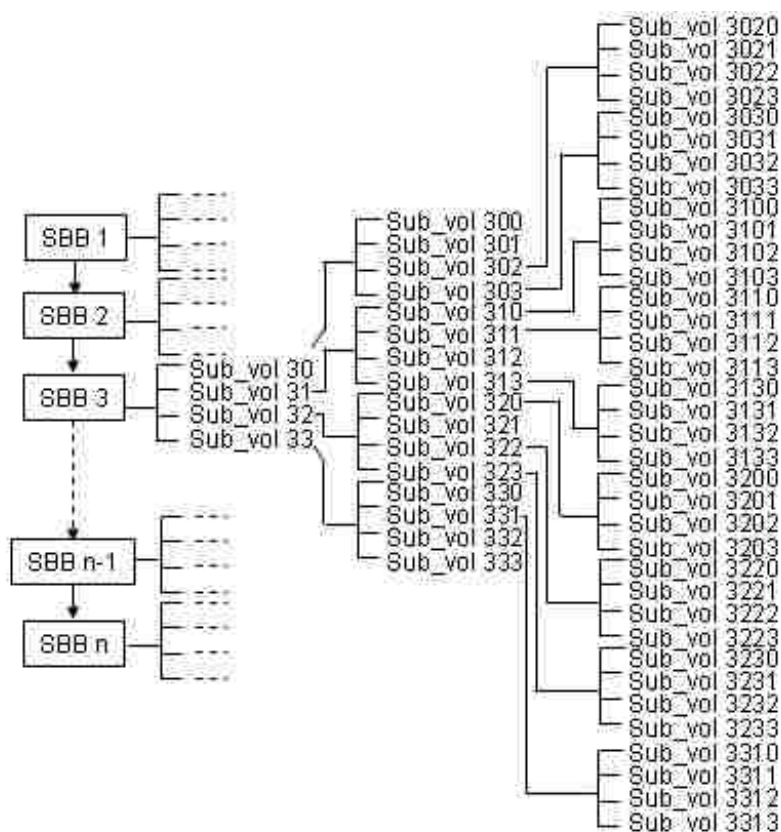


Figure 5.6. Data structure for the presented method

5.3. SOME RESULTS OF DATA MANAGEMENT

Several tests have been conducted to investigate the effectiveness of the proposed techniques of medical data management. One set of CT scan data (302 files with a total file size of 143 MB) is used for comparisons. A total of 14 files are created from image data processing with a bounding volume divided into 14 sub-volumes before the simulation. They are used as the new input data for bone drilling simulation, as shown in Table 5.1. This table illustrates that after using the proposed method, including first the OBB technique and then the SBB technique, the actual data files are just 8.133 MB in size, which is only 3.33% of the original data size. The ratio of the number of voxels of the actual data over the original data is only 2.12%. The processing time is only 1.89% using the actual data in comparison with using the original data. Compared to the single bounding volume data (i.e., without dividing into sub-volumes) shown in Figure 5.3(a), the file size and voxel number have been reduced to 31.42% and 31.35%, respectively. These comparisons are given in Table 5.2. Given these results, it is clear that the original data size and the processing time have been reduced drastically by implementing the proposed method. Note that the processing time is the sum of the data reconstruction time and the graphic rendering time.

Table 5.1. Comparisons among the original data, object bounding box (OBB) data, and sum of section bounding boxes (SBB Sum) data

File Name	File Size (KB)	Voxel Number	Processing Time(s)
Original Data	243,901	72,069,396	136.24
OBB	25,883	4,869,460	5.18
SBB Sum	8,133	1,526,400	2.16

Table 5.3 compares the number of sub-volumes between the uniform division and the quadtree subdivision for the 14 sub-volumes of the bone model. The table also shows the ratio of sub-volumes between these two techniques, indicating the memory savings

generated by the quadtree subdivision. The quadtree subdivision can save more memory as the number of times of subdivisions increase, since “full” quadrants do not need to be further subdivided and “empty” quadrants can be deleted from the data set. During the simulation, the number of voxels in the active sub-volume needs to be small enough to achieve real-time performance of the simulation system. It can be seen that the larger the number of subdivisions, the smaller the average number of voxels per sub-volume. The new “empty” quadrants in each subdivision can be deleted to save more memory during the simulated material removal process.

Table 5.2. Various ratios among the original data, object bounding box (OBB) data, and sum of section bounding boxes (SBB Sum) data

	File Size	Voxel Number	Processing Time
SBB Sum/Original	3.33%	2.12%	1.89%
OBB/Original	10.61%	6.76%	3.81%
SBB Sum/OBB	31.42%	31.35%	41.62%

Table 5.3. The numbers of sub-volumes for different numbers of subdivisions resulting from the uniform division and the quadtree subdivision and their ratio

No. of Subdivisions	Uniform Division	Quadtree Subdivision			Avg. Number of Voxels per Sub-volume	Ratio of Sub-volumes
		Full	Partially Full	Empty		
0	14	0	14	0	109,029	100%
1	56	0	56	0	27,257	100%
2	224	56	159	9	6,814	95.98%
3	896	56+134	334	9+168	1,704	58.48%

The optimal voxel number problem has been discussed in the literature. Held et al. [1995] stated that an optimal voxel number could be between $5 \times 5 \times 5$ and $50 \times 50 \times 50$ (where each of the three numbers refers to the number of divisions in each of X, Y, and Z

directions). In an experiment using a PC (P4 2.8G, 512M RAM), it can be found that when the average voxel number per sub-volume is around 30,000, real-time performance can be achieved. Note that the processing time for the subdivision is usually small enough that it does not affect real-time graphic rendering during the simulation

5.4. DATA AND MODEL UPDATE DURING TOOL-BONE INTERACTION

During the process of simulating material removal, when the virtual tool intersects active sub-volume(s) represented by “partially full” or “full” quadrants at the maximal level (e.g., subdivision number = 3 in Figure 5.5) of the tree, Boolean operation is directly conducted on those quadrants. When the virtual tool intersects the “full” quadrants on the internal level of the tree, the “full” quadrant becomes a “partially full” quadrant because of the material removal. Further subdivision is done until it reaches the preset subdivision number. Later Boolean operations are then performed on those newly created sub-quadrants. To further reduce memory and increase efficiency, the empty sub-volumes can be removed from the memory after each Boolean operation. The data structure, shown in Figure 5.6, is then updated through the deletion of existing sub-volumes and generation of new sub-volumes according to tool-bone intersections during the surgery simulation.

During the bone surgery simulation, the data changes continuously when the virtual tool removes material from the bone model and the updated data is re-rendered for graphic display. Instead of dealing with the entire volume, only the sub-volumes of interest are dealt at each time step. Typically there is only one active sub-volume. However, depending on the sub-volume size, the virtual tool size, and the tool position and orientation, multiple sub-volumes may exist at any time instant. A sub-volume queue is used to manage the tool-bone intersection, using pushback and popfront to temporarily store the sub-volumes intersected by the tool. The brief illustrations shown in Figure 5.7 are for two cases. In case a) the virtual tool’s volume is smaller than the smallest sub-volume and in case b) the virtual tool’s volume is bigger than the smallest sub-volume. Because the active sub-volume queue is shared between two independent threads, a runtime error will occur if the critical data are not handled well. To solve this problem, a synchronization technique called semaphore is used during implementation.

In addition to the interaction between the virtual tool and the virtual bone on the sub-volume level during simulation, the scalar fields also need to be modified on the voxel level for graphic re-rendering. The voxels examined during the walkthrough can be distinguished into three types in a 2D case, as shown in Figure 5.8:

Visited voxels: inside the world bounding block, but outside of the local bounding block, they can be quickly rejected.

Computed voxels: inside the local bounding block, but outside the tool's cutting volume. To identify these voxels, it is necessary to compute the tool's potential value for this point.

Modified voxels. inside the active cutting volume, the properties of these voxels need to be modified after applying the tool.

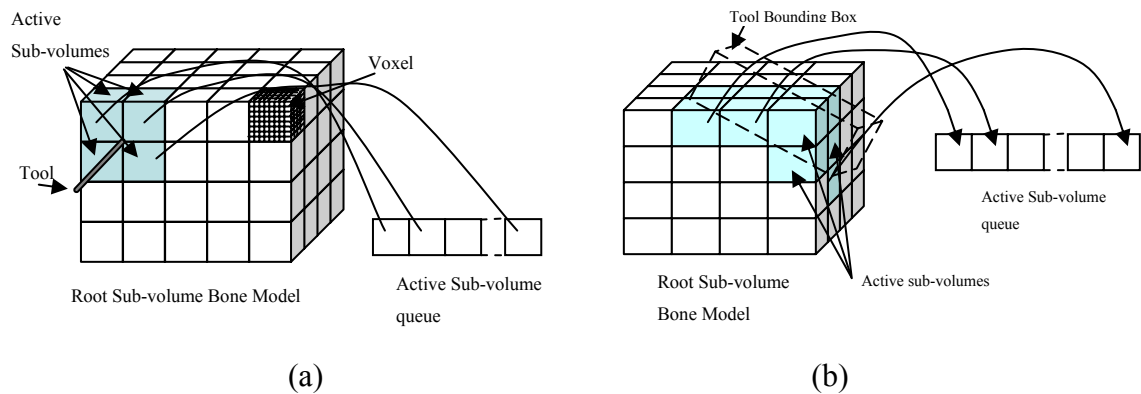


Figure 5.7. Example of tool-bone interaction and active sub-volume queue

During tool-bone interaction, World Bounding Block is checked first for the active sub-volume(s) of the virtual bone, and only the voxels inside the World Bounding Block will be checked for data model update. The active sub-volumes are pushed into the active sub-volume queue for later graphic re-rendering. The positions of the voxels in these sub-volumes need to be transformed to local coordinate system, and then they are checked whether they are inside the Local Bounding Block or not. If not, they are quickly

rejected as Visited voxels. Otherwise, these voxels are computed as Computed voxels using the virtual tool's implicit functions. The voxels inside the Active Cutting Volume are called Modified voxels, and their properties are modified for later graphic re-rendering.

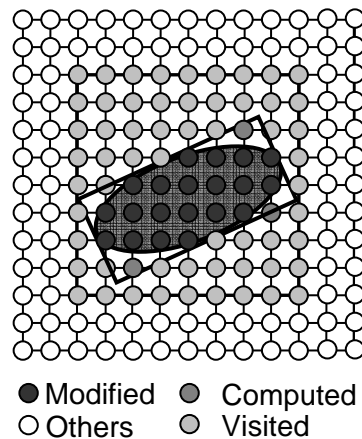


Figure 5.8. Voxel walkthrough during tool-bone interaction

At the same time, the segmentation data are changed in accordance with the modified voxels. After applying these three kinds of different volumes and computing three types of voxels, the local update are performed very well and the computational time can be saved significantly. Also, this method can be used for any kinds of virtual cutting tool.

5.5. ADAPTIVE RESOLUTION IMPROVEMENT

No matter which kind of machining processes are applied to the virtual bone surgery system, memory consumption and resolution problems are the two most challenging issues in volumetric modeling. It is obvious that the higher the resolution, the more the computer memory is needed. Much attention needs to be paid to how to balance these two needs, especially with the CT scans as input. The CT scans supplied by Depuy

Orthopaedics have a resolution of $0.48\text{mm} \times 0.48\text{mm} \times 1.5\text{mm}$ for X, Y, and Z direction respectively. This resolution is used for a volumetric cell that has eight voxels during data reconstruction. The results of graphic rendering, haptic rendering, and sound rendering are directly related to the resolution. Usually this resolution is fine enough for most applications, but it is not good enough for the finer rendering that will be necessary in the future research.

Although there are several ways to improve the resolution, one solution is to add more voxels to the existing volumetric bone model. The newly added voxels can be interpolated through zero-order interpolation (nearest neighbor function), first-order interpolation (tri-linear interpolation), or inverse distance weighted interpolation, depending on the balance of real-time performance and cutting surface quality. Note that the medical data management is going further, from the sub-volume level to the sub-voxel level, as shown in Figure 5.9.

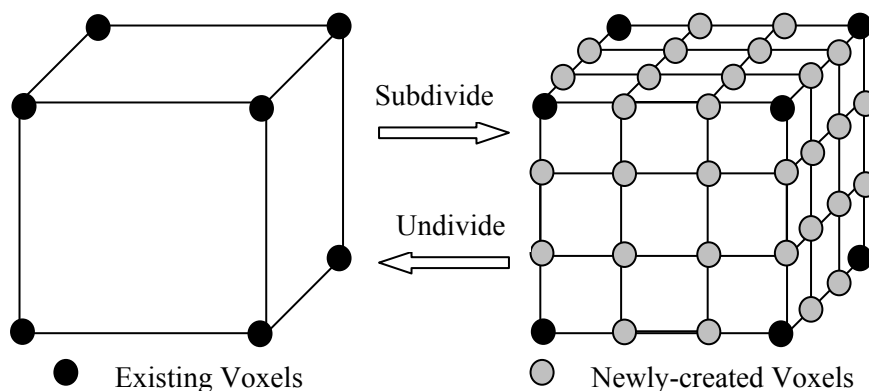


Figure 5.9. Sub-volume level to sub-voxel level

Based on our existing data management methods, two adaptive tool-guided resolution improvement methods, shown in Figure 5.10, are presented. In both of these methods, the active sub-volumes are resampled when the tool and bone interact. The difference is which portion of the active sub-volume is resampled. Method 1 is one-time

resampling for the whole interactive sub-volume, as shown in Figure 5.10(a). When the tool and the sub-volume interacts for the first time, the whole sub-volume is resampled and new voxels are created using the interpolation function, then the material removal process is conducted. At the next time step, no further resampling is needed; only the material removal process is conducted. Method 2 is more “local”. For each time step when the virtual tool interacts with the active sub-volume, resampling is conducted only at the intersection area of the tool and the bone, then material removal process is conducted, as shown in Figure 5.10(b). The data structure shown in Figure 5.6 is modified slightly for the new multi-resolution model.

These two methods both have advantages and disadvantages depending on parameters such as the tool position, orientation, shape and size, and the dimensions of the active sub-volume. Table 5.4 shows the comparison between these two methods. During implementation, method 2 is chosen for the resolution improvement.

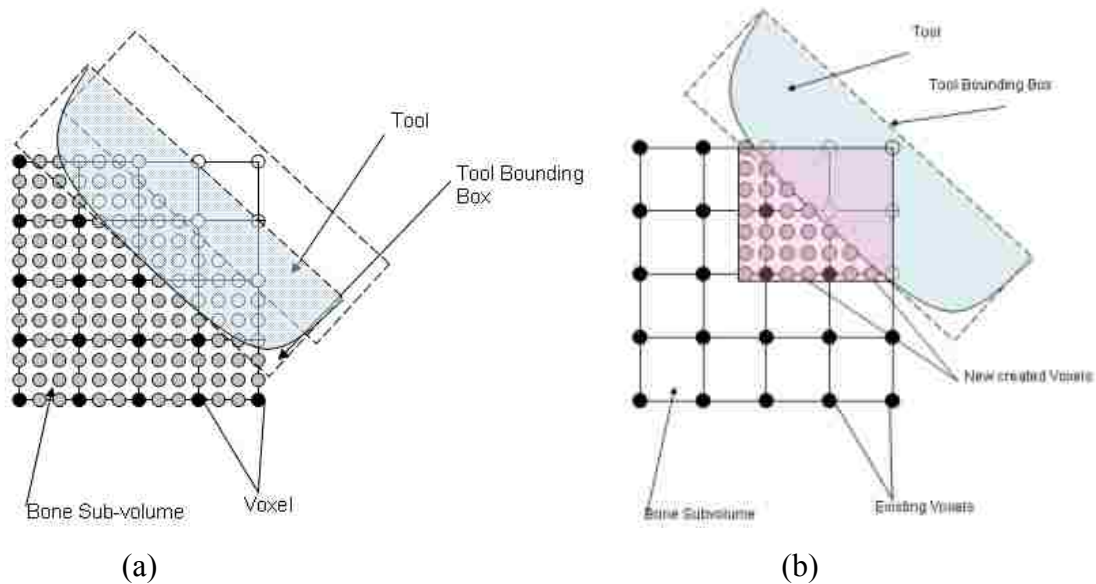


Figure 5.10. Adaptive tool-guided resolution improvement methods

Table 5.4. Methods comparison for resolution improvement

Comparisons	Method 1	Method 2
1	One time interpolation for all	Each time step needs interpolation
2	One time increase the memory size	Has less memory storage problem
3	Resolution can be fine	Resolution can be very fine
4	No further data structure needed	Need further data structure
5	Easy to search, implement	Need handle multiple-unconnected areas

With the resolution improvement, better display of graphics, haptics, and sound can be obtained. Figure 5.11 shows some results of graphic display for resolution improvement: (a) shows an example of graphic display for drilling on original resolution, (b) shows an example of the resolution increased 2 times; (c) shows an example of the resolution increased 4 times. To balance the visual realism and the run-time rendering speed, LOD (Level of Detail) will be considered and selected in different regions for graphic display purposes. A run-time algorithm will be developed for automatically select data of different resolution to guarantee an interactive rendering speed while maximizing model accuracy and rendering quality.

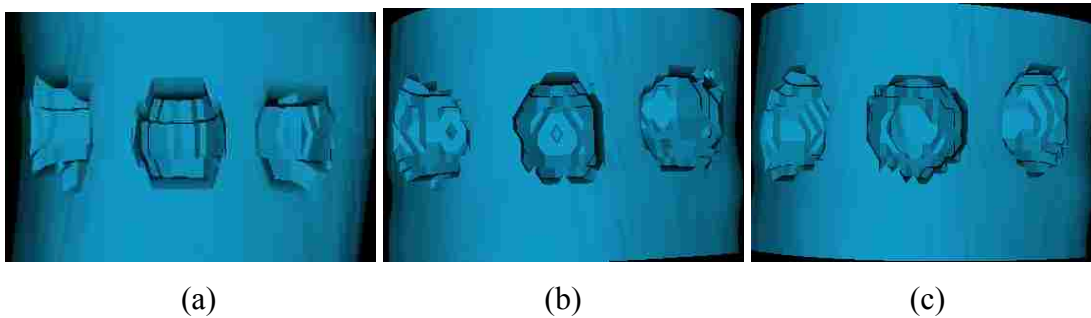


Figure 5.11. Examples of resolution improvement for graphic display

6. MODELING AND REPRESENTATION OF SURGICAL TOOLS

6.1. SURGICAL TOOL'S REPRESENTATION

Surgical tool's representation is essential to collision detection, force generation, haptic rendering, auditory rendering, and bone model and data update. A good representation will enable fast calculation, accurate material removal, realistic force feedback, and reasonable sound display for the virtual system.

For many of the existing virtual sculpting systems [Galyean and Hughes, 1991; Bærentzen, 1998] and bone surgery simulators [Agus et al., 2002; Peng et al., 2003; Morris et al., 2004], only a sphere-end cutting tool was used, with a sphere function to represent the virtual tool. It is not easy to introduce other kinds of virtual tools with different shapes in their systems. Some other sculpting systems used a volumetric method to model the tool. For example, Wang and Kaufman [1995] modeled carving tools using a volume-sampling technique, and stored the representation data in a volume raster of $20 \times 20 \times 20$ resolution. This representation method is easy for Boolean operations, and the small number of voxels reduces computations. However, they are associated with resolution and aliasing problems. In a virtual bone surgery system, several different machining processes need to be simulated, such as drilling, sawing, broaching, milling, etc. For each process, different cutting tools of various sizes, shapes and cutting areas are used. The virtual tool's representations mentioned above can not satisfy these requirements; therefore a better representation is needed for the bone surgery simulation.

In our VR system development, the surface of a virtual tool can be represented using implicit functions as follows:

$$S = \{(x, y, z) \in R^3 \mid F(x, y, z) = 0\} \quad (6.1)$$

where F is the implicit function and (x, y, z) is the coordinate of a point in 3D space. The implicit function separates a 3D space into three distinct regions as shown in Figure 6.1: inside, on or outside the defined surface by evaluating if $F(x, y, z) < 0$, $F(x, y, z) = 0$ or $F(x, y, z) > 0$, respectively.

For some complex shapes, primitive solids (spheres, cubes, and cylinders, etc.) and half-spaces can be combined into a hierarchical organization with the use of Boolean operations. For example, as shown in Figure 6.2, the half sphere representing a ball-end burr for a virtual reamer can be represented as the Boolean intersection of a half-space (A: $G(x, y, z) \leq 0$) and a sphere (B: $H(x, y, z) \geq 0$), i.e., $C = A \cap B$. For a point (x, y, z) in the 3D space, if $H(x, y, z) = 0$ and $G(x, y, z) < 0$, then this point is on the virtual reamer's surface; if $H(x, y, z) < 0$ and $G(x, y, z) < 0$, then this point is inside the virtual reamer's surface; otherwise, this point is outside the virtual reamer object. With implicit functions and half-spaces, it is easy to perform collision detection and determine if a voxel of a bone model needs to be removed during tool-bone interaction simulation.

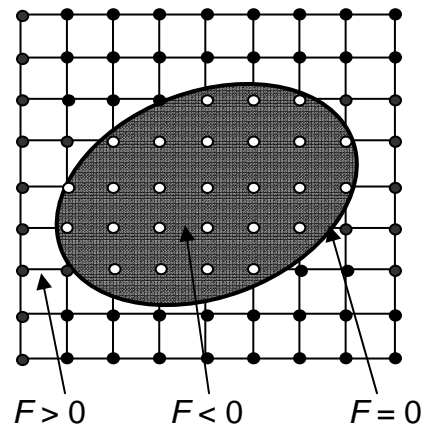


Figure 6.1. Three regions of implicit function

The surface normals of an implicit surface can be obtained using the gradient of the implicit function as follows:

$$\vec{n} = \frac{\nabla F}{\|\nabla F\|}, \text{ where, } \nabla F = \left[\frac{\partial F}{\partial x}, \frac{\partial F}{\partial y}, \frac{\partial F}{\partial z} \right] \quad (6.2)$$

In order to quickly locate the active sub-volume(s) and locally update the voxels of the virtual bone, the following terms are defined for the virtual tool to save calculation time as shown in Figure 6.2:

Active Cutting Volume: the actual cutting volume of the virtual tool. Implicit functions and their Boolean operations are used to represent this volume of the virtual tool as described in Section 2.1. These functions are represented in the virtual tool's local coordinate system.

Local Bounding Block: a local rectangular block that bounds the active cutting volume in the virtual tool's local coordinate system.

World Bounding Block: a global rectangular block, an axis-aligned bounding block (AABB), which bounds the local bounding block at any position and orientation in the system global coordinate system.

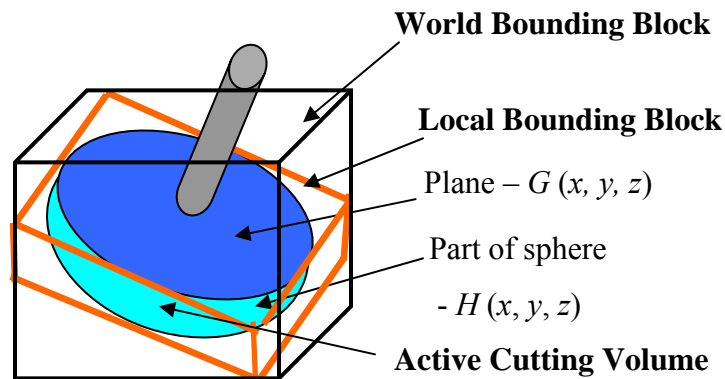


Figure 6.2. An example of reaming tool representation

Based on the multi-point collision detection, a virtual tool is represented by a number of sample points $P_i(x, y, z)$ distributed on the tool surface. Every point has an associated normal vector $\vec{n}_i(n_x, n_y, n_z)$ that is pointing to the inside of the tool. A set of

the inward pointing vectors \vec{n}_i and the sample points P_i describe the tool's shape. The sample points include passive points (on the cutter shank) and active points (on the effective cutting part). Figure 6.3 shows two examples of the tool representation using this technique. Active points exist only when the machining process is on; otherwise, all the points are passive points.

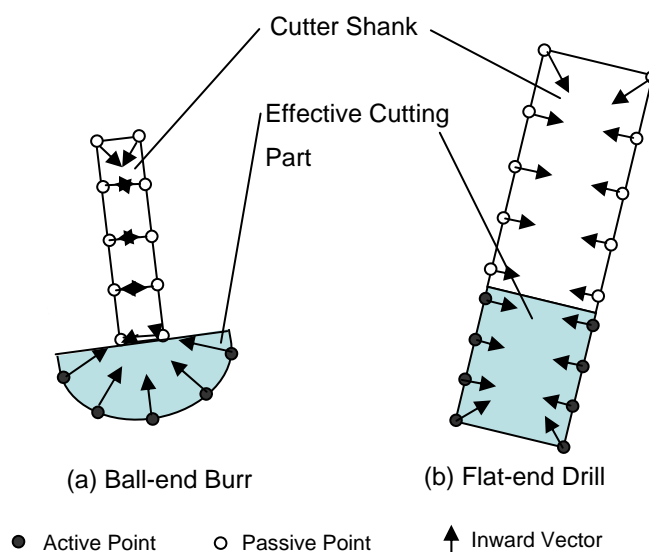


Figure 6.3. Examples of sample points on the tool's surface

In addition to cutting tools, there are also a variety of implants and other non-cutting tools, such as guides and suckers, which can be used in real bone surgery. For example, in temporal bone surgery the dominant hand usually controls the main tool, e.g. drills, while the contralateral hand controls a combination of irrigation/suction. In total knee replacement surgery, most of the machining processes will be guided by the special designed guides to make the position more accurate, as shown in Figure 6.4.

To create geometric models of all kinds of cutting tools and non-cutting tools, in the virtual bone surgery simulation system, the following procedures are performed:

1. Tool models are constructed using any CAD software, e.g., Unigraphics.
2. Tool models are exported as STL format files and saved onto a hard drive.
3. STL models are transformed into polygonal model with triangle meshes in the simulation system.
4. Special pipelines are designed and executed for display and calculation.

A tool box including main cutting tools and other tools, as well as implants has been designed. Some of them are shown in Figures 6.4 and 6.5.

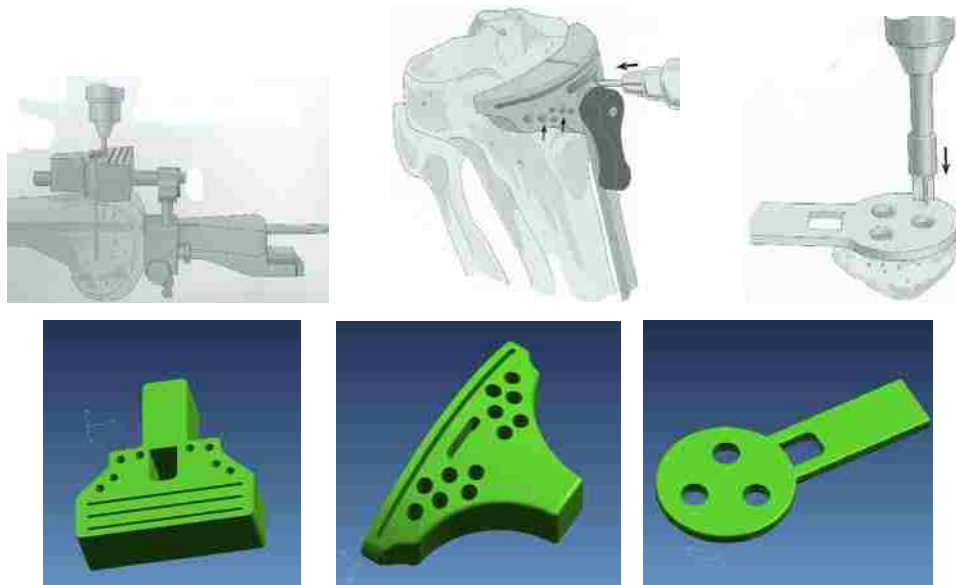


Figure 6.4. Guides modeling for total knee replacement

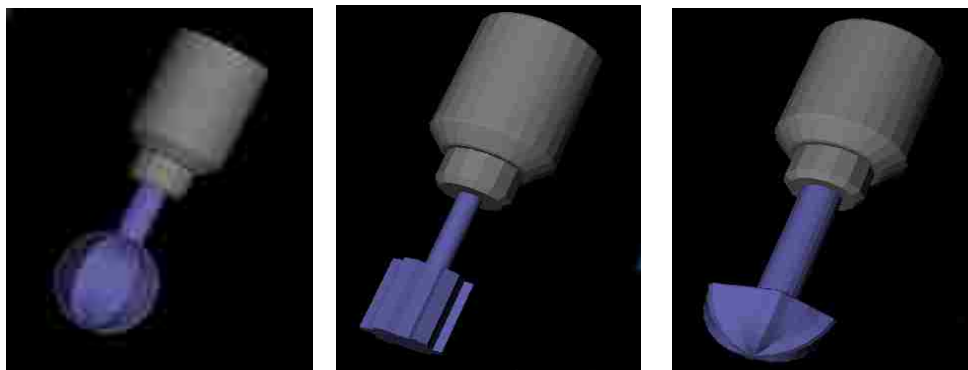


Figure 6.5. Some geometric models for cutting tools

6.2. VIRTUAL TOOL RELATED GEOMETRICAL CALCULATION

In order to find the geometric relationship among virtual bone, cutting tools, non-cutting tools, and inner holes, it is necessary to calculate their exact positions and orientations for every time step during the simulation in the virtual environment. Haptic devices usually provide six degree-of-freedom positional sensing, dx , dy , dz in translation for X -, Y -, Z - axis, and α , β , γ in rotation about X -, Y -, Z - axis.

Most haptic devices, e.g., PHANTOM, are single point based, thus geometrical transformation is needed for almost all the sample points on the virtual tool shown in Figure 6.6. Given a point $P = [x, y, z, 1]^T$ with the output positions and orientations of the haptic device, it is easy to get a new point P^* using translation matrix T_p :

$$T_p = \begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6.3)$$

and the rotation matrix R , which depends on the order of rotations (based on the visualization system's mechanism):

$$\begin{aligned} R &= R_z(\gamma)R_x(\alpha)R_y(\beta) \\ &= \begin{bmatrix} \cos \gamma & -\sin \gamma & 0 & 0 \\ \sin \gamma & \cos \gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha & 0 \\ 0 & \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \beta & 0 & \sin \beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \beta & 0 & \cos \beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos \gamma \cos \beta - \sin \gamma \sin \alpha \sin \beta & -\sin \gamma \cos \alpha & \cos \gamma \sin \beta + \sin \gamma \sin \alpha \cos \beta & 0 \\ \sin \gamma \cos \beta + \sin \alpha \cos \gamma \sin \beta & \cos \gamma \cos \alpha & \sin \gamma \sin \beta - \sin \alpha \cos \gamma \cos \beta & 0 \\ -\cos \alpha \sin \beta & \sin \alpha & \cos \alpha \cos \beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \quad (6.4)$$

All virtual tools can be moved freely in the virtual working environment and some of the non-cutting tools, e.g., the guides shown in Figure 6.4, can also be placed on the virtual bone. In order to predict the geometrical relationship between the cutting tool and the non-cutting tool (see the virtual drill and a virtual guide shown in Figure 6.6 (a)), all sample points on the drill and the guide's position need to be known. The projection of the guide and the sample points can be used for prediction, but it will cost a lot of computation time. To simply the calculation, the coordinate transformation shown in Figure 6.6(b) and simple calculations for the drill can be conducted.

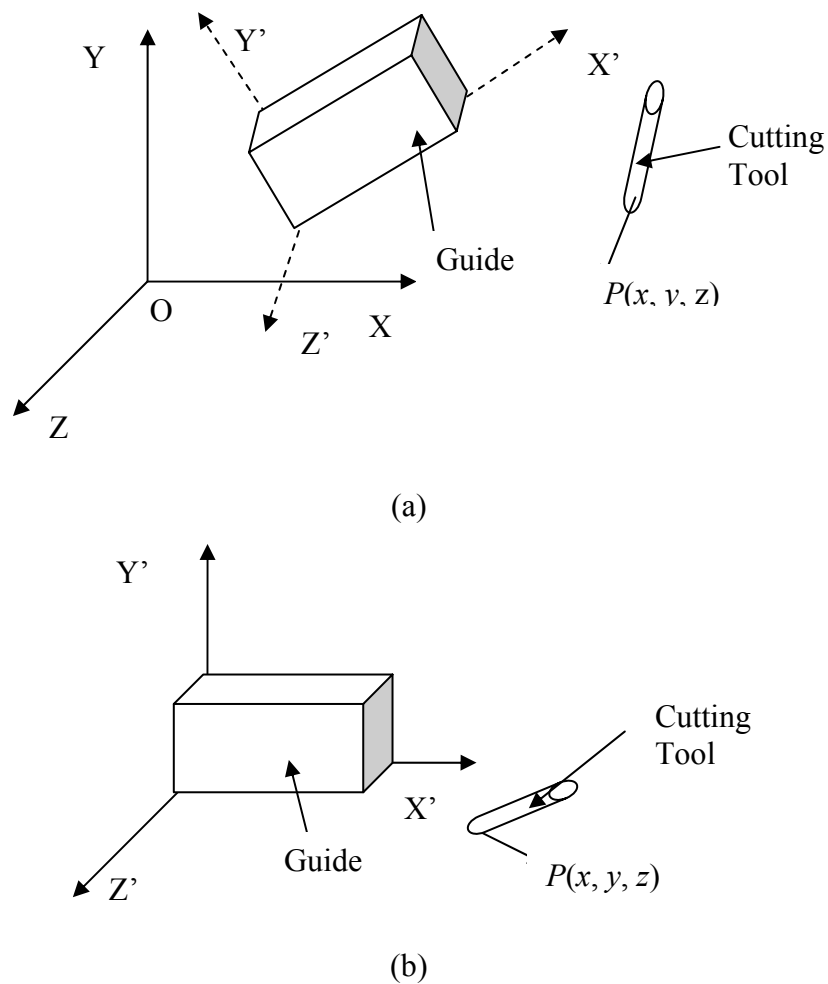


Figure 6.6. Simplified calculation for the relationship between guide and tool

For one of the sample points on the drill, e.g., $P(x, y, z)$, the calculation for the point P' in the coordinate system $X'Y'Z'$ is

$$P' = PT^{-1} \quad (6.5)$$

where T is the geometrical transformation matrix for the virtual guide, and can be

presented as a 3×3 matrix $\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$ for faster calculation.

$$T^{-1} = \frac{T^*}{|T|} \quad (6.6)$$

where $T^* = \begin{bmatrix} a_{22}a_{33} - a_{23}a_{32} & -a_{12}a_{33} + a_{13}a_{32} & a_{12}a_{23} - a_{13}a_{22} \\ -a_{21}a_{33} + a_{23}a_{31} & a_{11}a_{33} - a_{13}a_{31} & -a_{11}a_{23} + a_{13}a_{21} \\ a_{21}a_{32} - a_{22}a_{31} & -a_{11}a_{32} + a_{12}a_{31} & a_{11}a_{22} - a_{12}a_{21} \end{bmatrix}$ and

$|T| = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31} - a_{11}a_{23}a_{32} - a_{12}a_{21}a_{33}$ (Sarrus Rule).

With the newly calculated coordinates of the given point, it is easy to determine whether it is inside or outside the virtual guide because the virtual guide can also be represented using implicit functions.

7. FORCE MODELING AND HAPTIC RENDERING

7.1. INTRODUCTION

Haptic interface can enhance the realism of virtual surgery by giving a realistic feel for the surgical operation. Haptic rendering is the process of applying reactive forces to the user through a force-feedback device [Okamura, 1998]. The rendering uses of using information about the tool-object interface to assign forces to be displayed, given the action at the operational point. The major challenge in simulating force-reflecting volume models is to achieve an optimal balance between the complexity of geometric models and the realism of the visual and haptic displays.

The following issues must be addressed in order to provide meaningful force feedback [Peng et al., 2003; Hua and Qin, 2002]:

1. Force computation rate must be high enough and the latency must be low enough in order to generate a proper feel of the operation.
2. Generation of contact force creates the feel of the object during the simulated surgery. Interaction forces between the tool and the bone can be calculated using mathematical models.

For haptic rendering, there are several important components, as shown in Figure 7.1: force modeling, collision detection, force computation, and haptic rendering.

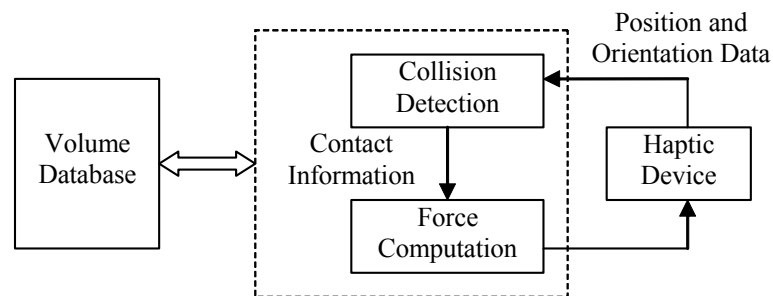


Figure 7.1. Structure of haptic rendering

7.1.1. Force Modeling. Bone material removal operations are of considerable importance in orthopedic surgery [Plaskos et al., 2003]. In hip and knee replacement procedures, for instance, the geometrical accuracy of the prepared bone surface is particularly relevant to achieving accurate placement and good fixation of the implant.

Bone drilling is necessary prior to many orthopedic operations, such as pin or screw insertion to the bone, and requires a high level of surgical skills. Several studies on bone drilling were reported in the literature. Wiggins and Malkin [1976] investigated the interrelationships between thrust pressure, feed rate, torque, and specific cutting energy (energy per unit volume required to remove material) for three types of drill bits. Jacob et al. [1976] presented research results showing that the drill point geometry was critical when attempting to minimize drilling force and that a softening effect occurred when the bone was drilled at relatively high speeds. Hobkirk and Rusiniak [1977] studied the relationships between drilling speeds, operator techniques, types of drills and applied forces in bone drilling.

Through experiments they showed that the peak vertical force exerted on the drill varied between 5.98 N and 24.32 N, and that the mean vertical force ranged from 4.22 N to 18.93 N. By testing the drilling force against the bone hardness and triaxial strength, Karalis and Galanos [1982] found a linear correlation between the triaxial compressive strength and the drilling force. Abouzgia and James [1995] investigated the dependance of force on drill speed and measured an energy consumption during drilling. They found that drilling force increased slightly with increase in speed at low starting speeds and decreased with increase in speed at high starting speeds.

Allotta et al. [1996] developed an experimental model for the description of breakthrough during the penetration of a twist drill in a long bone, as illustrated in Figure 7.2. They presented an equation for the thrust force required to drill a hole and reported its good correlation with experimental data. The thrust force required to drill a bone is

$$T = K_s a \frac{D}{2} \sin \frac{\beta}{2} \quad (7.1)$$

where T is thrust force, K_s is total energy per unit volume, a is feed rate expressed in unit length per revolution, D is diameter of the drill bit, and β is convex angle between the

main cutting lips (see Figure 7.2). K_s represents the sum of shear energy required to produce gross plastic deformation. It is primarily the friction energy of the chip sliding past the tool plus other minor energies. K_s has been shown to vary between $4.8R_u$ and $6R_u$ [Allotta et al., 1996], where R_u is the unitary ultimate tensile load. $K_s = 5R_u$ is a practically acceptable value. During rotation and penetration through the bone, the drill bit is subject to the resistant torque (in addition to the thrust force) of

$$M_z = 5R_u a \frac{D^2}{8} \quad (7.2)$$

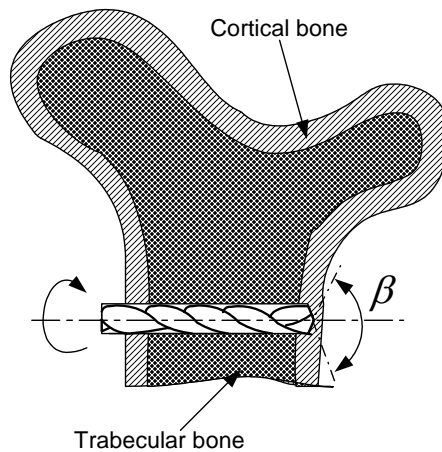


Figure 7.2. Modeling force in drilling a long bone

Chi et al. [2005] presented another drilling force model by performing regression of measured drilling force versus process and material parameters. The obtained force model was validated by performing more experiments with different sets of parameter values. The thrust force model could be written as:

$$T = 134.6N^{-0.3327}v^{0.5189}\rho^{1.1841} \quad (7.3)$$

where T represents the thrust force, N is speed of drill bit in rotations per minute, v is feed-rate in mm/sec, and ρ is bone material density in g/cc.

Bone burring is also an important surgical procedure used in temporal bone surgery. Agus et al. [2002] presented a bone-burr interaction model. For a burr with a spherical bit of radius R rotating at angular velocity ω , they used Hertz's contact theory to derive the following elastic deformation force that exerts on the burr:

$$\vec{F}_e = C_1 R^2 \left(\frac{h}{R} \right)^{\frac{3}{2}} \hat{n} \quad (7.4)$$

where C_1 is a constant that depends on the elastic property of material, h is the tool embossing height, \hat{n} is the normal direction of the contact surface. Also, the friction force can be obtained using

$$\vec{F}_\mu = \mu \int_{\xi} P(\vec{\xi}) \frac{\vec{r}(\vec{\xi}) \times \vec{\omega}}{|\vec{r}(\vec{\xi}) \times \vec{\omega}|} d\sigma \quad (7.5)$$

where μ is a friction coefficient, $\vec{\xi}$ represent a point on the contact surface, $P(\vec{\xi})$ is the pressure exerted by the burr on point $\vec{\xi}$, $\vec{r}(\vec{\xi})$ is the displacement measured from the center of sphere burr bit to point $\vec{\xi}$, and $d\sigma$ represents a differential area on the contact surface.

The total force that should be provided by the haptic feedback device is, therefore,

$$\vec{F}_T = \vec{F}_e + \vec{F}_\mu \quad (7.6)$$

Other force models can also be applied in developing a virtual bone surgery system. For example, Eriksson et al. [2005] used an energy-based approach to determine how force is as a function of material removal rate during the milling process. This model

is same as the following simplified milling force model [Yang and Chen, 2003; Choi and Jerard, 1998]:

$$F_t = K_t(MRR) / f \quad (7.7)$$

where F_t is the tangential cutting force, f is the feedrate, and MRR is the material removal rate. The radial force is

$$F_r = K_r F_t \quad (7.8)$$

where K_t and K_r are constant and their values depend on workpiece material, cutting tool geometry and cutting conditions.

The spring-damping force model [Hua and Qin, 2002; McNeely et al., 1999; Avila and Sobierajski, 1996] can also be applied to virtual bone surgery. The force in this model can be expressed as

$$F = R(\vec{V}) + S(\vec{N}) \quad (7.9)$$

where $R(\vec{V})$ is the damping force, which is a motion retarding force proportional to velocity. $R(\vec{V})$ can be written as

$$R(\vec{V}) = -t_r(\rho, |\nabla\rho|)\vec{V} \quad (7.10)$$

where \vec{V} is the velocity of the contacting point and t_r is a function of density ρ . $S(\vec{N})$ is the stiffness force normal to the object surface and is

$$S(\vec{N}) = \frac{\vec{N}}{\|\vec{N}\|} t_s(\rho, |\nabla\rho|) \quad (7.11)$$

where \vec{N} is the surface normal at the contacting point and t_s is a function of density ρ . For volume rendering and isosurface rendering, t_r and t_s can be expressed differently [Hua and Qin, 2002; Avila and Sobierajski, 1996].

A haptic device can be used to give virtual bone surgery system users a realistic force feedback by rendering the force and torque computed using cutting force models. At present, most virtual bone surgery systems use the PHANToM device (SensAble Technology Corp.) and General Haptics Open Software Toolkit (GHOST) SDK for haptic rendering. This PHANToM device has three motors and six encoders to enable 6-DOF motion tracking and 3-DOF force feedback. The GHOST SDK is a C++ object-oriented software toolkit that allows application developers to interact with the haptic device and create a virtual environment at the object level. GHOST SDK provides a special class of functions called `gstEffect`, which allows adding “global” forces directly to the PHANToM. At each iteration of a servo loop the pointer of the Effect object is passed to a PHANToM node. By generating the Effect force when non-null intersection between the virtual tool and the virtual bone is detected, the system gives the user a realistic feel of force in real time.

A multithreading virtual environment can be implemented in order to run the components of a virtual surgery system asynchronously. The multithreading computation environment allows maintaining suitable update rates for the various subsystems of the simulation system. The haptic loop must maintain an update rate of about 1,000 Hz, while the graphics loop can get by with an update rate of about 30 Hz.

7.1.2. Collision Detection and Force Generation. In a bone surgery simulator, the haptic rendering consists of two parts: collision detection and force generation. The goal of collision detection, also known as interference detection or contact determination, is to automatically report a geometric contact when it is about to occur or has just occurred [Lin and Gottschalk, 1998]. Fast and accurate collision detection between geometric models is a fundamental problem in computer based surgery simulation. In developing a virtual bone surgery system, it is necessary to perform collision detection for the purposes of simulating material removal and force feedback.

An early approach to haptic rendering used single-point representation of the tool for collision detection and penalty-based methods for force generation [Massie and

Salisbury, 1994; Avila and Sobierajski, 1996]. Collision detection was done by checking whether the point representing the tool was inside the object of consideration such as a bone. The surface information of an anatomic model can be obtained in terms of triangular facets using the marching cube algorithm previously described or by a method of surface reconstruction from dixel data [Peng et al., 2004].

Penalty-based methods generate a pre-computed force field based on the shortest distance from the interior point of an object to the object's surface. Figure 7.3 shows the problems of penalty based haptic rendering. One problem with this approach is that there may be points in an object which have the same distance to the surface (see Figure 7.3(a)). Another problem is that when pressing an object with a sharp tip or fine feature, such as the one shown in Figure 7.3(b), the user will quickly feel the change of force direction from one side of the object to the other side and then feel no force at all. This can be a serious problem, especially when working with highly detailed models and small structures.

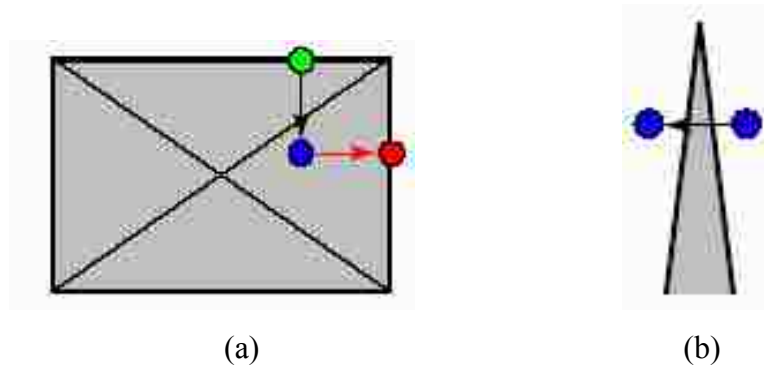


Figure 7.3. Problems of penalty based haptic rendering

Constraint-based methods were introduced by Zilles and Salisbury [1995] and by Ruspini et al. [1997, 1998]. These methods use an intermediate object (representing the tool) which never penetrates a given workpiece, such as a bone in the environment, as shown in Figure 7.4. The intermediate object (called God-Object or Proxy) remains on the surface of the workpiece during the simulation process. The force generated by the

haptic device is proportional to the vector difference between the physical position of the virtual tool and the proxy position of the virtual tool. The haptic rendering algorithm updates the proxy position in respect to the physical position by locally minimizing the distance from the proxy position to the physical position. Since these calculations have to be performed on-the-fly, constraint-based approaches are computationally more expensive than penalty-based approaches.

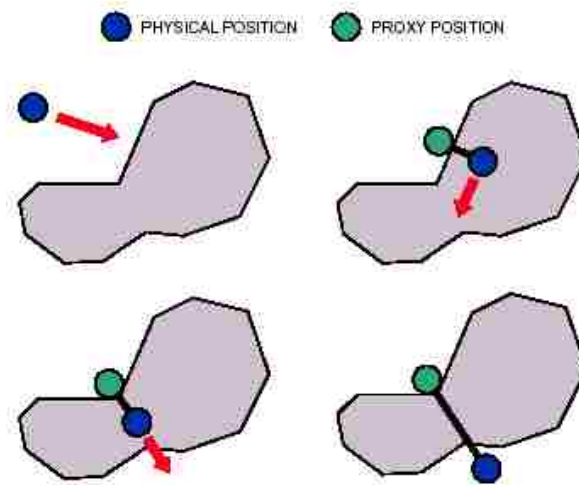


Figure 7.4. Haptic rendering by virtual proxy [Zilles and Salisbury, 1995]

The single-point representation of an object for collision detection, as described above, has many drawbacks:

1. It is not suitable for inhomogeneous workpiece material, e.g., human bone.
2. It does not represent the 3D shape of a virtual tool.
3. It has discontinuities problem, e.g. sharp edges on the surface.
4. The virtual tool can reach points which may not be reachable by the real tool, e.g., entering a small hole with a large tool [Niu and Leu, 2007].

Multi-point collision detection methods have been developed more recently [McNeely et al., 1999; Petersik et al., 2002]. These methods handle 3D shapes using

multiple points on the surface of the tool. Using these methods, more realistic simulations of tools and tool-object interaction can be achieved and the drawbacks of the single-point approach can be overcome. However, multi-point collision detection is computationally more expensive. Moreover, this force feedback scheme may generate an unstable force in some cases [Nakao et al., 2003], especially when the number of points on the tool surface is not adequate.

7.2. MULTI-POINT COLLISION DETECTION

As described in the previous section, both single point based collision detection and multi-point collision detection have advantages and disadvantages. In this research, Pointshell idea is extended and made the multi-point approach more suitable to our problem:

1. Make the points with inward vector laying on the tool's surface as illustrated in the previous section.
2. Apply new collision detection algorithm which is suitable for the tool-bone interaction.
3. Apply new multi-point based force calculation.

All these modifications help improve the computational accuracy of the method and make the simulation of tool-bone interaction more realistic. For inhomogeneous materials like a human bone, the multi-point approach can also supply information on the material distribution around the cutter. Furthermore, it only requires surface normal calculation for the tool and does not require the calculation for the bone. It is also worth mentioning that the larger the number of the sample points, the more accurate the result.

A key issue of multi-point collision detection is to determine whether a sample point is inside, outside or on the surface of the virtual object. Because the virtual bone is modeled directly from CT scan data in our work, the structured data type is used for volume buffer to represent the bone. Since volumetric modeling method is used for the virtual bone, there does not contain an explicit representation of the bone surface and the surface must be calculated using the voxel data's properties. To shorten the calculation, the medical image segmentation information for each voxels is used for collision detection. From the segmentation data acquired from the medical image processing

methods [Niu et al., 2008], every voxel inside the virtual bone has a value of 1 and every voxel outside the virtual bone has a value of 0. These segmentation data are stored as one of the properties of voxel data.

With the help of the concepts of voxel and volumetric cell, it is easy to find the 3D relationship between the virtual bone and the virtual tool. As shown in Figure 7.5, the multi-point collision detection algorithm is based on these relationships. For every sample point generated from the virtual tool, at each time step, collision is checked as follows. First, find the virtual bone's cell number for a given sample point. If all the voxels' segmentation data for this cell are marked as 1, the sample point is inside the virtual bone surface. If all the data are 0, then the sample point is outside the bone surface. If the voxels' segmentation data in that cell have both 0 and 1 values, the sample point may be on the bone surface depending on applying which kind of approximation method. For example, if a nearest point method is used for checking whether the sample point are on the surface, the accuracy of this collision detection will increase to half-voxel size comparing to regular collision detection. To further increase the accuracy, an interpolation for the eight neighbor voxels can be employed. Once one sample point is checked as on or inside the bone surface, the collision occurs, otherwise, the other sample points need to be checked. If all the sample points are checked, and no one is on or inside the bone surface, then no collision happens. The detailed collision detection algorithm is shown below:

```
[Algorithm: Collision Detection]
  For all given  $n$  sample point  $P_i$ ,
    Check the cell number  $C_i$  for  $P_i$ ,
    Can't find cell number, no collision, next sample point
    Found the cell number, check the 8 voxels' segmentation data of  $C_i$ 
      If all data are 0,
        no collision, next sample point
      else if all data are 1,
        collision happened, return true
      else
        find the  $P_i$ 's nearest voxel  $V$  in  $C_i$ 
        if  $V$ 's segmentation data is 1,
          collision happened, return true
        else
          no collision, next sample point
        end if
      end if
    No more sample point, return false
```

When the virtual cutting tool is off, all the two kinds of sample points are checked for collision detection; when the machine is on, the active points are also checked for material removal and force generation, which will be described next.

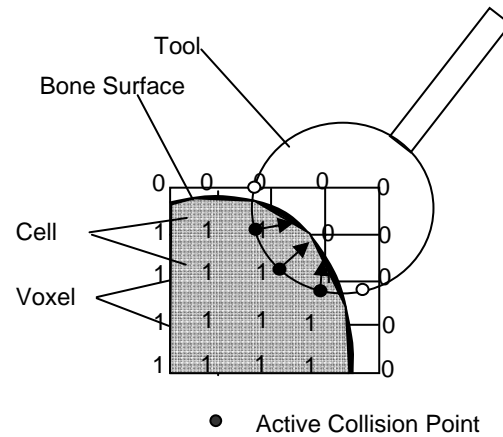


Figure 7.5. An example of multi-point collision detection

7.3. CUTTING FORCE CALCULATION

From the literature, some of the force models are related directly to the cutting tools, e.g., drill [Allotta et al., 1996; Udiljak et al., 2003]. Most existing research, both in the virtual sculpting field and the bone surgery simulation field [Agus et al., 2002; Peng et al., 2003; Morris et al., 2004], used simplified force models such as spring-damping force, viscous force, material removal rate, etc. In order to calculate the cutting force whenever a collision between the tool and the workpiece, the direction and magnitude of the force need to be calculated each collision point (P_i^C, \vec{n}_i^C) . However, most existing bone surgery simulation systems have not taken material properties around the cutting tool as considerable parameters for the force calculation of haptic rendering, and also some of them just use single point based force calculation, so the both the direction and

magnitude of the final force may not be right. Thus, to get a realistic haptic force for the tool-bone interaction, a multi-point based force calculation is presented.

To get the haptic force with both direction and magnitude, each active collision point (active point inside the virtual bone) needs to be computed with high precision. The force direction of each active collision point is same as the inward vector of this sample point as defined in previous section. The magnitude of this collision force can be calculated as proportional to the material removal rate, which is also proportional of the tool velocity. Considering the material properties of this point, the collision force can be specified as

$$\vec{F}_i = K_1 K_2 D_i \left\| \frac{\vec{P}_i^c - \vec{P}_i^l}{\Delta t} \right\| \vec{n}_i \quad (7.12)$$

where K_1 is force field stiffness on the surface, K_2 is a coefficient for the effective cutting area for each sample point, D_i is the bone density at this point, \vec{P}_i^c is the current position of the tool, \vec{P}_i^l is the previous position, \vec{n}_i is the surface normal at the active sample point, and Δt is the sampling period.

The total interface force between the tool and the bone is

$$\vec{F} = \sum_{i=1}^k \vec{F}_i \quad (7.14)$$

Figure 7.6 shows an example of force calculation for two consecutive time steps. The direction and magnitude of the total interface force gives a realistic result for the force calculation.

The multi-point based force calculation can supply more realistic haptic forces because the material properties around the cutting area are considered during the tool-bone interaction. However, any haptic device has a limited force which can be applied to it. When the user pushed harder, it is possible that the physical position of the tool

immerses completely into the virtual model. To overcome this limitation, a modified proxy object algorithm [Zilles and Salisbury, 1995] is implemented. Whenever a more than a certain number of surface sample points are in contact with the static object, the way back to the proxy is traced until the number of contacts is below the limit or until the proxy is reached. At that location the force vector is calculated as described above and the proxy is set to that position.

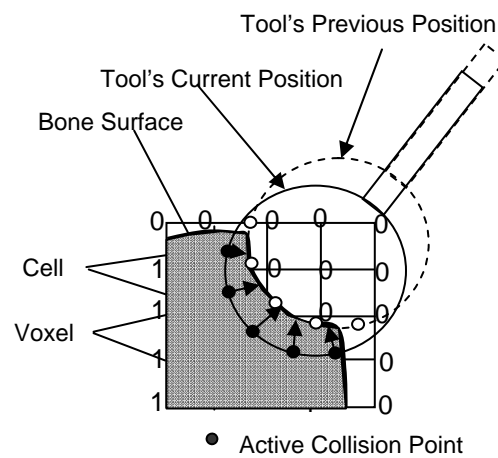


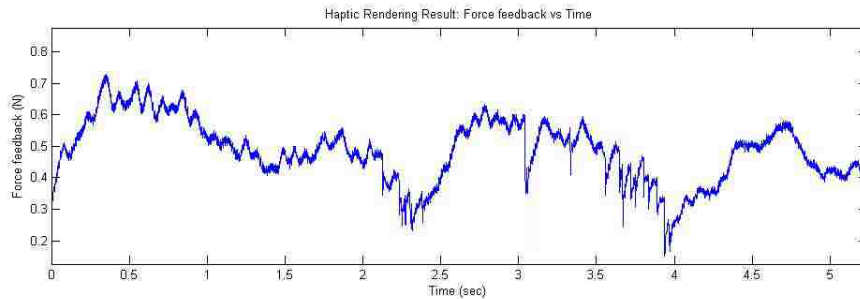
Figure 7.6. An example of force calculation for two consecutive time steps

Although the multi-point based model gives a valid solution for real-time computation, the force feedback scheme has a possibility of generating unstable force in some cases. Especially when the number of sample points is not adequate, the calculated force vector sometimes changes drastically because the velocity is calculated by differentiation of displacement. This is known to be not smooth and may have large variations. Therefore, the force stability technique must be applied to solve the problem of inconsistent force generation, hence improving the force fidelity. One of the force filtering techniques, an exponentially weighted moving average function [Peng et al., 2003] is used to filter force:

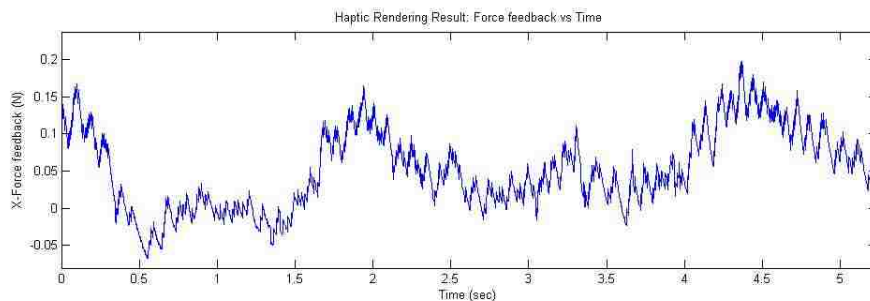
$$F_i = E_i(1 - \alpha) + F_{i-1}\alpha \quad (7.15)$$

where F_i is the expected force value at time step i , α is the smoothing factor and its value is between 0 and 1, and E_i is the value of the sampled function at time step i .

The user can select different kinds of surgical tools to practice different machining processes of bone surgery. Figure 7.7 shows the simulated surface burring force profiles. When the user continuously operates the virtual round burr to move the surface layer of the virtual bone, the force is applied to the haptic device according to tool's movement. With the depth of the burr below the surface level, the results of the haptic force are very close to the results reported by Agus et al. [2002]. Figure 7.8 shows the simulated milling force profiles.

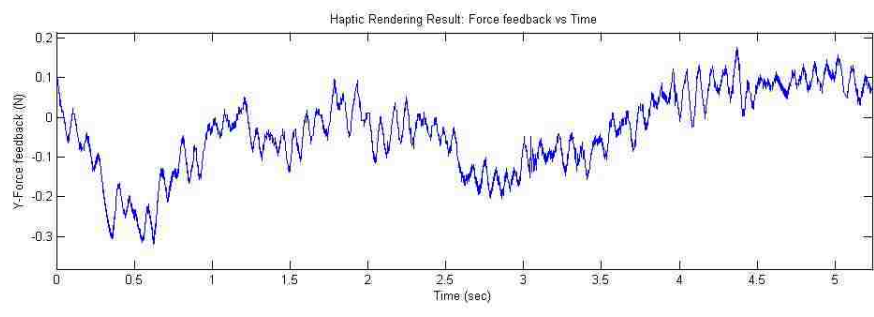


(a)

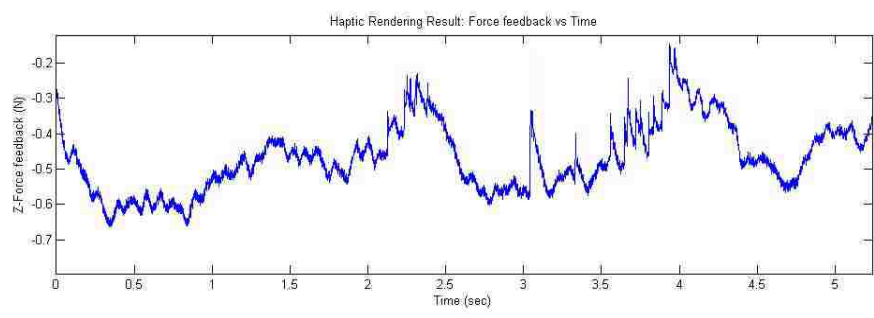


(b)

Figure 7.7. The simulated force profile for burring on cortical bone: (a) total force; (b)-(d) haptic force along X-, Y-, and Z-axis

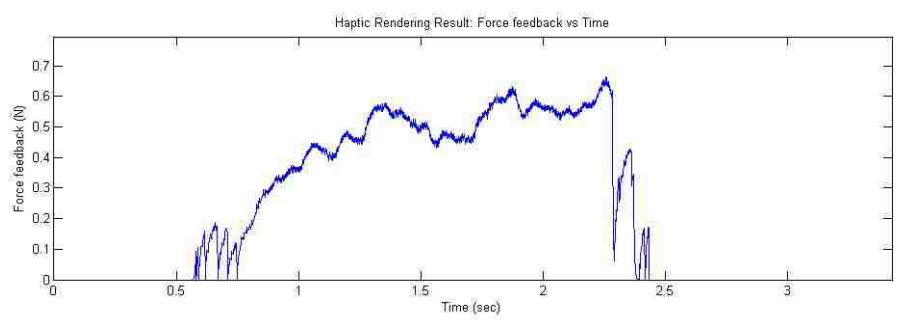


(c)



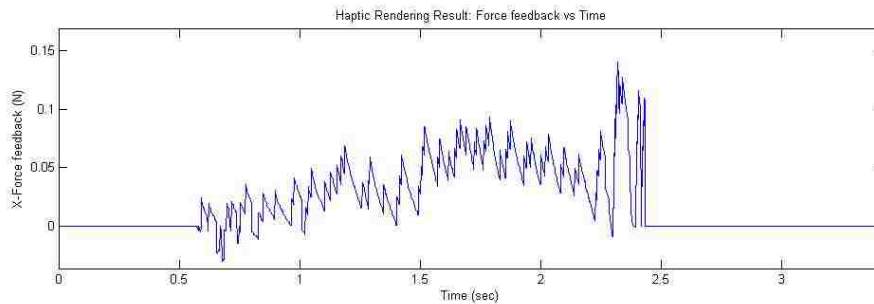
(d)

Figure 7.7. The simulated force profile for burring on cortical bone: (a) total force; (b)-(d) haptic force along X-, Y-, and Z-axis (cont.)

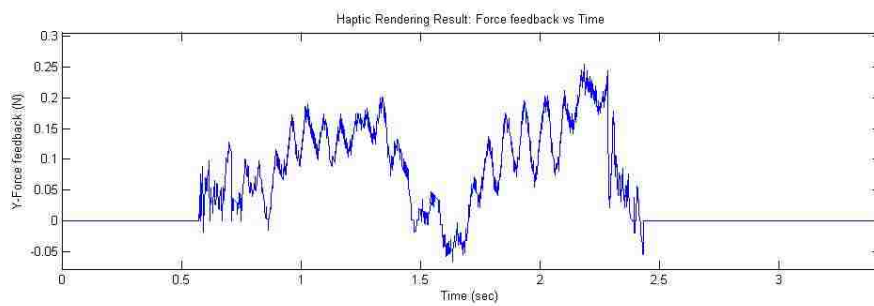


(a)

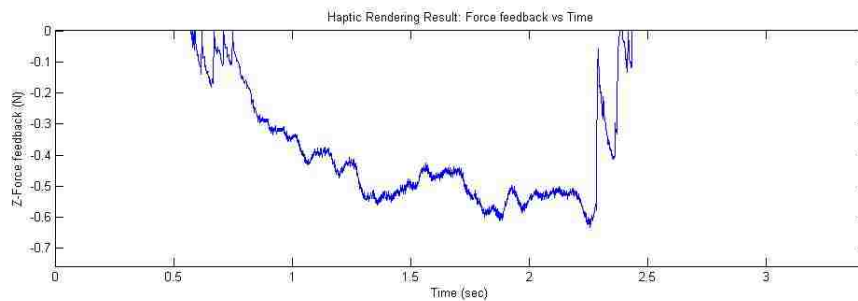
Figure 7.8. The simulated force profile for milling on cortical bone: (a) total force; (b)-(d) haptic force along X-, Y-, and Z-axis



(b)



(c)



(d)

Figure 7.8. The simulated force profile for milling on cortical bone: (a) total force; (b)-(d) haptic force along X-, Y-, and Z-axis (cont.)

8. SOUND MODELING AND RENDERING

8.1. INTRODUCTION

While human beings are generally considered sight-dependent creatures, there is no disputing the importance of auditory cues in our ability to relate to the environment. Sound plays an important role in real bone surgery. It provides information about tool contact and about the nature of the underlying bone. Experienced surgeons can know the progress and status of the surgical event even through the sound generated during material removal in a bone surgery operation. For example, the change of sound from higher to lower pitches in bone drilling could signal reaching the interface between the bone and the soft tissue. The surgeon could then stop drilling to avoid damage to the tissues, which will occur if the drill passes through the boundary of the bone. Therefore it is desirable to include sound in a virtual bone surgery system, so that the VR system can be enriched to a full multimodal interaction environment including auditory rendering besides graphics and haptics. Therefore the user can perform virtual bone surgery by simultaneously “seeing” bone material removal through a graphic display device, “feeling” the force via a haptic device, and “hearing” the sound of tool-bone interaction.

In a virtual reality system with sound rendering, two kinds of sounds are used: pre-recorded sound and synthesized sound. Most current VR systems use pre-recorded sound, which is easy to acquire and to play. But there are many drawbacks associated with using pre-recorded sounds [Miner and Caudell, US patent, 2004]. The sound is static and can not be changed in response to changes within a simulation environment including user interactions. Also, a large sound library is required to create a VR system with an acoustically rich virtual environment. Furthermore, it is difficult and impractical to obtain an application-specific sound sequence for every application. Synthesis sound, on the other hand, is flexible, dynamic, and especially good for the user action related virtual reality scenarios compared to pre-recorded sound. Thus in the virtual bone surgery system developed in this work, synthesized sounds is used to simulate a drill’s material removal during tool-bone interactions.

Although some research work can be found in the literature regarding the synthesis of contact sound for interactive simulation in a virtual environment [Pai et al.,

2001; van der Doel and Pai, 1998], there has been very little effort on sound synthesis for material removal. Most studies in virtual bone surgery concentrate on graphic and haptic interfaces, and only a couple of papers [Wiet et al., 2002; Morris et al., 2006] can be found in the literature about auditory rendering for virtual bone surgery. The auditory information of these studies are either incorrectly applied or not address the subtle change characteristics [Shine et al., 2006]. The most challenging issue of sound synthesis in virtual bone surgery is to get a sound model that allows real-time simulation while being sufficiently accurate to represent the important features of the sound during tool-bone interaction.

8.2. REVIEW ON SOUND MODELING METHODS

There has been some initial research work on the modeling synthesis of sounds for interactive simulation in a virtual environment. Basically these methods can be categorized into physical modeling synthesis and spectral modeling synthesis: Physics based methods employ the knowledge of the physical laws that govern the motions and interactions within the system under studying and expressing them as mathematical formulae and equations; spectral sound synthesis methods are based on modeling the properties of sound waves as they are perceived by the listener [Tolonen et al., 1998]

8.2.1. Physical Modeling Synthesis. Physical characteristics of objects involved can be modeled to the synthesized sound. The features of this approach are:

1. It is usually used in musical acoustics and computer music research;
2. It usually simulates physical behavior of an actual instrument;
3. It is not generalizable to many different types of sounds;
4. Sometimes it is too complex and is not suitable for real-time purpose.

Physical modeling methods can be divided into five categories [Valimaki and Takala, 1996]: numerical solving of partial differential equations; source-filter modeling; vibrating mass-spring networks; modal synthesis; digital waveguide synthesis.

Hiller and Ruiz [1971] were the first to take the approach of solving the differential equations of a vibrating string for sound synthesis purposes. The basic principle is to obtain mathematical equations that describe the vibratory motion in the object under study. These wave equations are then solved in a finite set of points in the

object, thus obtaining a differential equation. The use of differential equations leads to a recurrence equation that can be interpreted as a simulation of the wave propagation in the vibrating object. Chaigne and Askenflet [1992, 1994a, 1994b] also have been studying finite difference methods with applications of modeling of guitar, piano, violin and xylophone.

Source-filter models have been used especially for modeling the human sound production mechanism. The interaction of the vocal chords and the vocal tract is modeled as a feed forward system. Effective digital filtering techniques for source-filter modeling have been developed especially for speech transmission purposes. This technique has been used especially to produce synthetic speech [Moorer, 1985], but also for musical applications [Roads, 1995].

Cadoz et al. [1983] attempt to model the acoustical system under study of using simple ideal mechanical elements, such as masses, dampers and springs. They aim to develop a paradigm which can be applied to an arbitrary acoustic system, CORDIS. The CORDIS system was the first system capable of producing sound based on a physical model in real-time [Florens and Cadoz, 1991].

Modal synthesis is based on the premise that any sound-producing object can be represented as a set of vibrating substructures which are defined by modal data [Adrien, 1991]. Substructures are coupled and they can respond to external excitations. These coupling connections also provide for the energy flow between substructures. The simulation algorithm uses the information of each substructure and their interactions such as bodies and bridges, bows, acoustic tubes, etc.

Digital wave guide synthesis is one of the most widely used physics-based sound synthesis methods in use today. It is very efficient in simulating wave propagation in one-dimensional homogeneous vibratory systems, and the method is very much digital signal processing oriented [Smith, 1992]. It has been used for developing physical models of strings, wind and brass instrument and the human singing voice. However, the technique is not extensible to general sound synthesis.

8.2.2. Spectral Modeling Synthesis. Because spectral sound synthesis methods are based on modeling the properties of sound waves, they are much more general and can be applied to model a wide variety of sounds. Several spectral modeling synthesis

methods can be found in the literature [Valimaki and Takala, 1996]: traditional linear synthesis methods (additive synthesis, the phase vocoder, and source-filter synthesis); McAulay-Quatieri algorithm; spectral modeling synthesis (SMS); transient modeling synthesis; the inverse-FFT based additive synthesis method (FFT^{-1}).

Additive synthesis is a method in which a composite waveform is formed by summing sinusoidal components. It can be interpreted as a method to model the time-varying spectra of a tone by a set of discrete lines in the frequency domain [Smith 1991]. In additive synthesis, three control functions are needed for every sinusoidal oscillator: the amplitude, frequency, and phase of each component. The main drawbacks of the additive synthesis are the enormous amount of data involved and the demand for a large number of oscillators. The method gives best results when applied to harmonic or almost harmonic signals where little noise is presented [Tolonen et al., 1998].

The phase vocoder was developed at Bell laboratories and was first described by Flanagan and Golden [1966]. All vocoders present the input signal in multiple parallel channels, each of which describes the signal in a particular frequency band. Vocoders simplify the complex spectral information and reduce the amount data needed to present the signal. The phase vocoder works best when used with harmonic and static or slowly changing tones. It has difficulties with noisy and rapidly changing sound signals.

The McAulay-Quatieri (MQ) algorithm originated from research of speech signals but it was already reported in the first study that the algorithm is capable of synthesizing a broader class of sound signals [McAulay and Quatieri, 1986]. The original signal is decomposed into signal components that are resynthesized as a set of sinusoids. The method is efficient in presenting harmonic or voiced signals with little noise or transitions. If noisy or unvoiced signals are to be reproduced, a large number of sinusoids is needed.

The spectral modeling synthesis (SMS) technique was developed by Serra [1989] for decomposing a sound signal into deterministic and stochastic components. The deterministic component can be obtained by using the MQ algorithm [McAulay and Quatieri, 1986] or by using a magnitude-only analysis. The deterministic part is subtracted from the original signal either in the time or the frequency domain to produce a residual signal which corresponds to the stochastic component. However, the SMS

model restricts the deterministic part to sinusoidal components with piecewise linear amplitude and frequency variations, and the stochastic component is described by its power spectral density. Serra used Short-Time Fourier Transform (STFT) to analyze sounds, so this approach is not easily parameterizable or extensible due to limitations of the STFT.

Verma et al. [1997] extended the spectral modeling synthesis to a transient modeling synthesis (TMS). In this approach, the residual signal obtained by subtracting the sinusoidal model from the original signal is represented in two parts, the transients and steady noisy components. TMS is based on the duality between the time and the frequency domains. Transient signals are impulsive in the time domain, but with a suitable transformation, impulsive signals are presented as frequency domain signals that have a sinusoidal character. This implies that sinusoidal modeling can be applied in the frequency domain to obtain a parametric representation of the impulsive signal.

Inverse FFT (FFT^{-1}) synthesis is presented in [Rodet and Depalle, 1992]. In this method, additive synthesis is used in the frequency domain, i.e., all the signal components are added together as spectral envelopes composing a series of STFT frames. The waveform can be constructed by calculating the inverse FFT of each frame. The overlap-add method is used to attach the consecutive frames to each other.

Besides two main categories of modeling synthesis methods motioned above, there are also some other methods like FM and AR found in the literature. FM (Frequency Modulation) synthesis, originally introduced by Chowning [1973], is a fundamental digital sound synthesis technique employing a nonlinear oscillating function. It combines two or more sinusoidal waves to form more complex waveforms. AR (Auto Regressive) modeling was used by Kim et al. [2005] to simulate small drill sound for dental simulator. This mathematical modeling of a time series is based on the assumption that each value of the series depends only on a weighted sum of the previous values of the same series plus noise. The linear models give rise to rapid and robust computations.

8.3. METHODS OVERVIEW FOR SOUND MODELING AND RENDERING

The mechanism of sound generation in the bone material removal process is highly complex. It may relate to the bone and tool vibrations excited by the dynamic

interface force, which in turn depends on the force excitation, bone and tool materials, part dimensions, and process parameters. Briefly described, the primary objective of sound modeling and rendering for virtual bone surgery is to play the drilling sound for the tool-bone interactions in response to user actions. Thus, the proposed methods consist of the methods of sound acquisition in the real world, sound analysis and modeling for breaking the pipeline, sound characteristics analysis, mathematical model generation, and sound rendering in virtual bone surgery system for auditory display from virtual environment to the real world. The block diagram is given in Figure 8.1.

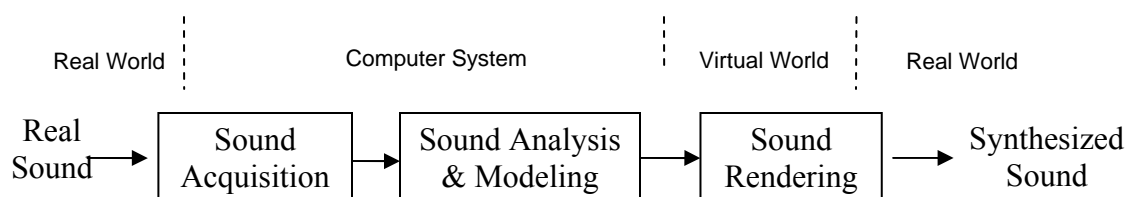


Figure 8.1. Block diagram of sound modeling and rendering

Usually there are two ways to recreate the sound wave field at both ears of the listener: using headphones (binaural techniques) or using loudspeakers (transaural techniques). Thus two kinds of sound output for different virtual environment setup are supplied for the virtual bone surgery simulation system. The user can choose a headphone or a loudspeaker for a virtual bone surgery practice. In real world, surgeons are quite adamant that they would prefer to avoid any further encumbrance such as earphones or headphones, polarized goggles or HUD's in addition to what they are already obliged to wear in the O.R., thus in this research, a loudspeaker is chosen as the main sound output for sound rendering and a headphone is only used when the user does not want to disturb the others during a private self-practice.

Although there are many methods for sound synthesis modeling, there has been very little work on sound synthesis associated with material removal [Shine et al., 2006].

For the objective of this research, it is very hard to use physical-based method to model the machining sound because the mechanism of sound generation in the bone material removal process is highly complex. Thus a spectral synthesis modeling method is presented for sound modeling for virtual bone surgery simulation, and details will be given at Section 8.5.

When synthesizing the sound, several parameters need to be considered [Doel and Pai, 1998]:

1. The shape of the object – represented by the frequency response;
2. The location of the impact – represented by the relative amplitudes of the frequency components;
3. The material of the object – represented by the decay of frequency component due to internal friction;
4. The force of the impact – the amplitude of the emitted sound was proportional to the square root of the energy of the impact.

If related these parameters to the bone material removal process, sound generation may associate to the bone and tool vibrations excited by the dynamic interface forces, which in turn depend on bone and tool materials, part dimensions, and process parameters such as spindle speeds, feed rates of the cutter, etc.

8.4. SOUND ACQUISITION

Because of the complexity of machining sound and the large number of parameters involved, sound analysis, modeling, and rendering is based on an experimental framework. The recorded sound will be used as the input for later sound analysis, modeling and rendering. Sound acquisition is conducted to capture bone machining sound, and to convert the captured sound to PCM (Pulse Code Modulation) wave files for later analysis, modeling, and synthesis. According to loud speaker-based transaural filtering or headphone-based binaural filtering, the experiment is setup as shown in Figure 8.2.

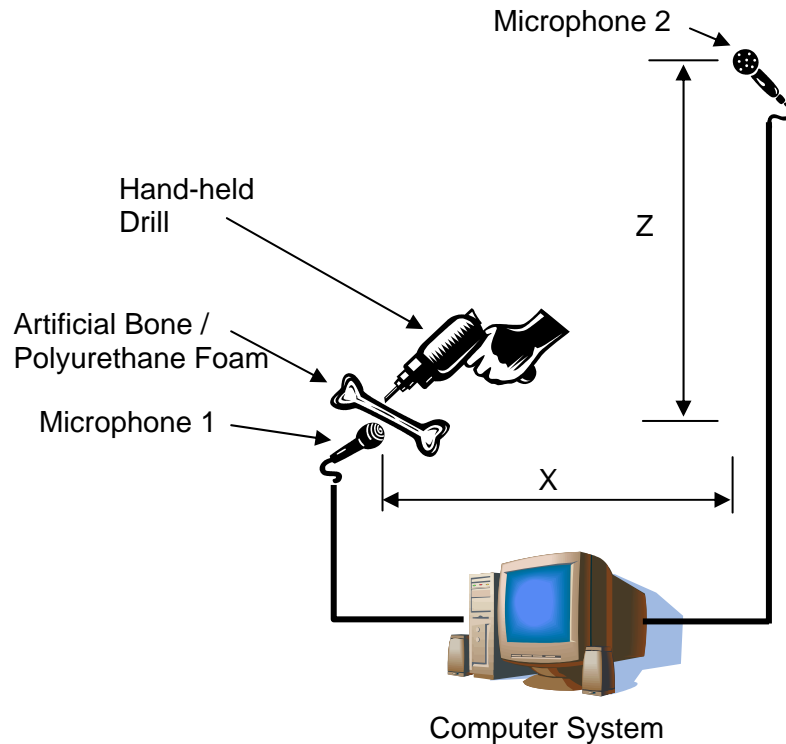


Figure 8.2. Schematic of experiment for sound acquisition

Microphone 1 is used to record the machining sound near where the tool-bone interaction happens. The sound captured by Microphone 1 is the primary sound output using a loudspeaker for sound rendering after sound synthesis modeling. Microphone 2 is located at the approximate position of the surgeon's ear during real surgery. X could be chosen as 20-25 mm and Z could be chosen as 35-45 mm. The sound captured by Microphone 2 is used for later sound modeling and rendering of headphone output in the case of private self-practice.

Hayes et al. [1997] presented that alternative materials with similar mechanical features of human bone could be used for bone related experiments in lieu of human or animal bone tissue. Thus the experiments are performed on some artificial bone material sheets called SAWBONES[®] from Pacific Research Laboratories. These material sheets are primarily used in testing of orthopaedic implants, instruments and instrumentation as an alternative testing medium to human cadaver bone, because they offer uniform and

consistent physical properties such as density, (compressive, tensile, and shear) strength and (compressive, tensile, and shear) modulus based on ASTM (American Society for Testing and Materials) F-1839, ASTM D-1621, ASTM D-638 and ASTM D-695. [<http://www.sawbones.com>]. The material sheets used in our experiments, which represent different human bone materials, consist of the following:

- Solid rigid polyurethane foam is primarily used as an alternative test medium for the cancellous / trabecular bone of a human.
- Cellular rigid polyurethane foam is also used as an alternative test medium for the cancellous / trabecular bone of a human. Its outward appearance matches cancellous bone more closely than does the solid rigid polyurethane foam.
- E-Glass filled epoxy sheet is primarily used as an alternative test medium for human cortical bone. It is a mixture of short E-Glass fibers and epoxy resins that have been pressure molded into thin sheets.
- 3rd Generation Composite Humerus is a type of artificial human bone having typical material properties including both simulated cortical bone (E-Glass filled epoxy) and simulated cancellous / trabecular bone (rigid polyurethane foam).

The major typical material property considered in the experiment is the density. Polyurethane foam blocks with density of 10 pcf (pounds per cubic foot), 20 pcf, 30 pcf, and 40 pcf are used to represent cancellous human bone in testing. For cortical bone, E-Glass filled epoxy sheet with density of 106 pcf is used. 3rd Generation Composite Humerus is used for the testing of bone drilling-through. Figure 8.3 shows these different testing materials.

A standard 1/8" drill bit and a hand-held power drill are used in the experiments. Two Samson C01U USB studio condenser microphones are used for sound recording.

The LabView software is used to record and convert the captured sound for different material sheets and drilling parameters. A sample collecting frequency is chosen to follow Nyquist sampling rate and human hearing frequency. To avoid aliasing occurring in the sampling of a signal, the sampling rate should be greater than or equal to twice the highest frequency present in the signal. As the whole frequency spectrum that can be heard by a human ear is 20 Hz to 20 kHz, a sampling rate of 44 kHz is good

enough for data collection. The resolution of the captured sound is set as 16 bits, mono, and the maximum measuring time sequence is around 10 seconds considering the large amount of data the sound measurements produce.

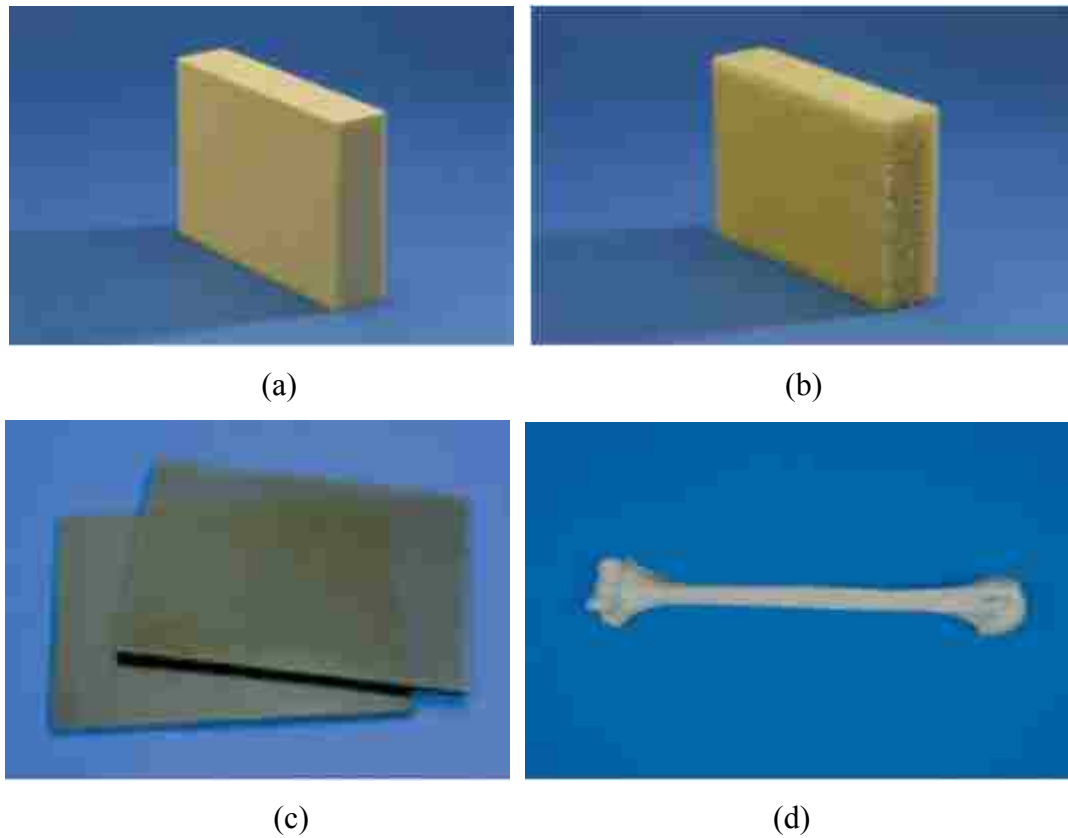


Figure 8.3. Experiment testing materials: (a) solid rigid, (b) cellular rigid polyurethane foam, (c) e-glass filled epoxy sheets and (d) 3rd generation composite humerus

Each testing material is drilled with several holes as trials with empty-run drilling prior to real drilling. These captured sounds are saved as PCM wave file format, and then they can be used for later sound analysis and sound modeling.

8.5. SOUND ANALYSIS

In order to recognize the characteristics of the drilling sound for later auditory rendering, captured sound data are analyzed both in time domain and frequency domain. Spectral subtraction is performed to obtain the sound difference caused by tool-bone interaction.

8.5.1. Sound Analysis in Time Domain. Sound analysis and modeling are based on captured sounds which are converted to PCM (Pulse Code Modulation) wave files during sound acquisition stage. It is important to follow the standard file format in order to read the captured sound wave sound as input for analysis/modeling, and to write the synthesized sounds in time domain back to wave files. Figure 8.4 shows the PCM wave file format. Some important attributes for reading/writing PCM wave file are: NumChannels describing mono or stereo sound; Sample Rate for the sampling rate (Hz); BitsPerSample for the sound sample resolution; data containing raw sound data.

After reading in the captured PCM wave files, it is easy to obtain the drilling profile which shows the magnitude versus time in time domain. For the drilling sound profile in time domain, the amplitude is the major parameter for analysis. Figure 8.5 shows the comparison of drilling sound on cortical bone material (106 pcf) and drilling on cancellous bone material (40 pcf). The difference in the amplitudes of these two drillings is obvious. From above observations, it is possible to classify the sound signals at different states by looking at the amplitude in time domain.

There are several different drilling states and transitions for bone drilling. These states and transitions can be revealed using sound analysis in time domain and also using spectrogram. Figure 8.6 shows a typical drilling sound displayed with different drilling states and transitions on cortical bone material in time domain. These states include: before drilling, free-running, drilling in bone material, drilling out bone material (perforation) and after drilling, it is very clear that the amplitude of the drilling in bone material state is much larger than the amplitude of the state of free-running, since amplitude represents the power of a signal. The greater the amplitude, the greater the energy carried. So the energy in the drilling in bone material state is much more than that in the free-running state and drilling out bone material (perforation) carries the maximum energy among all the states, even though the period is very short.

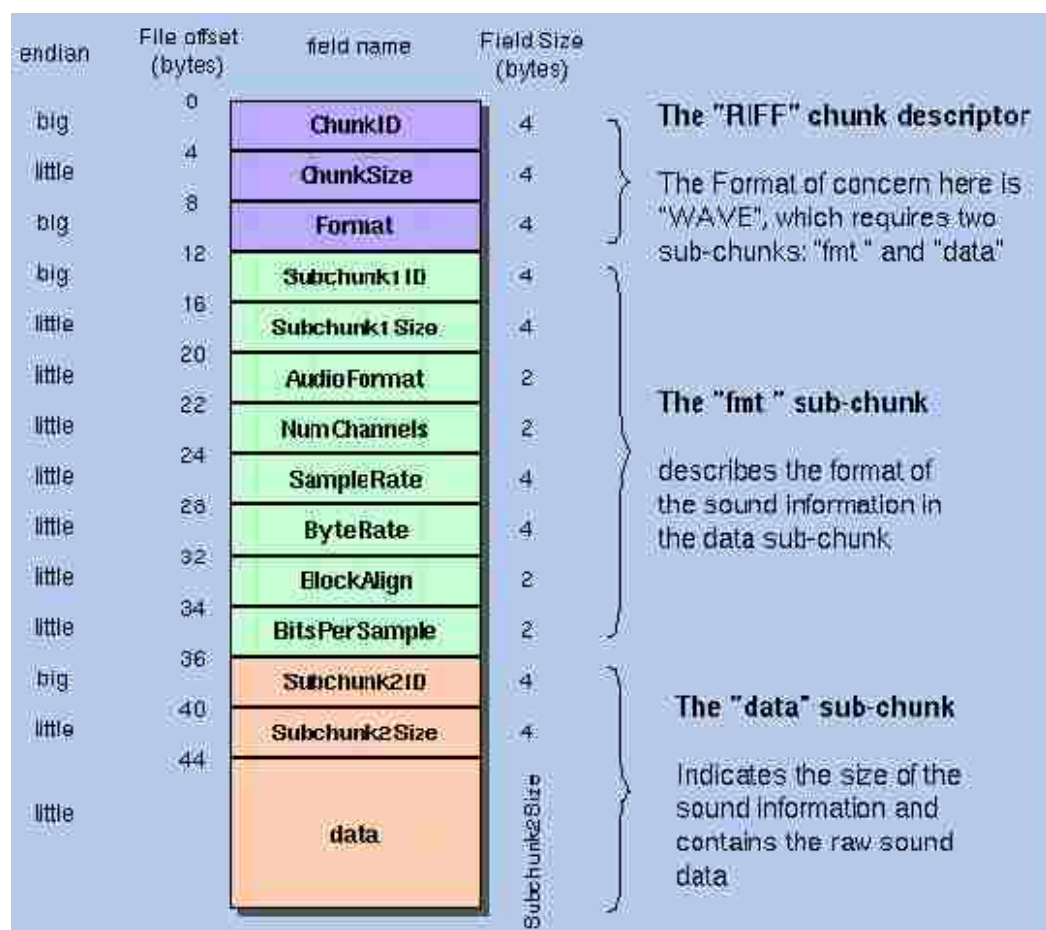


Figure 8.4. PCM wave file format

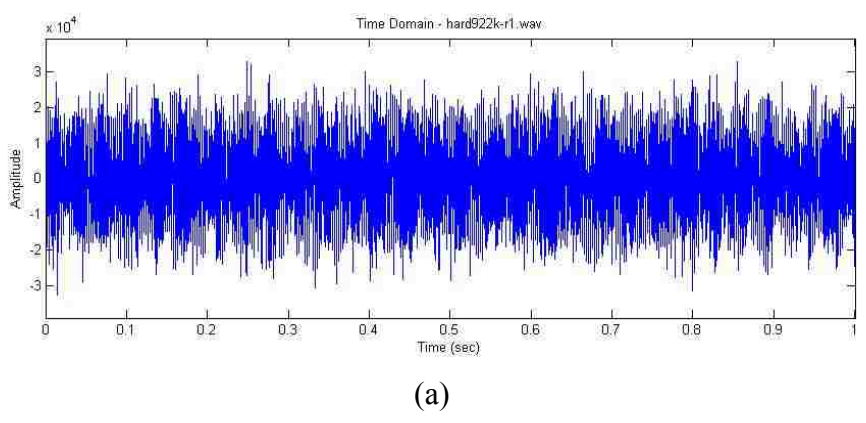
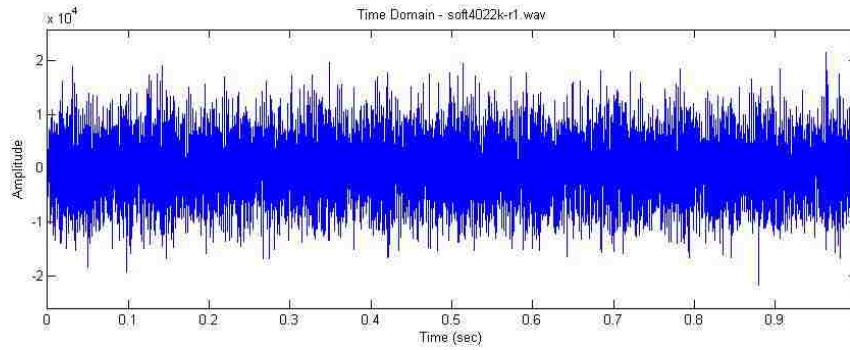


Figure 8.5. Drilling on different materials in time domain: (a) cortical bone material drilling; (b) cancellous bone material drilling



(b)

Figure 8.5. Drilling on different materials in time domain: (a) cortical bone material drilling; (b) cancellous bone material drilling (cont.)

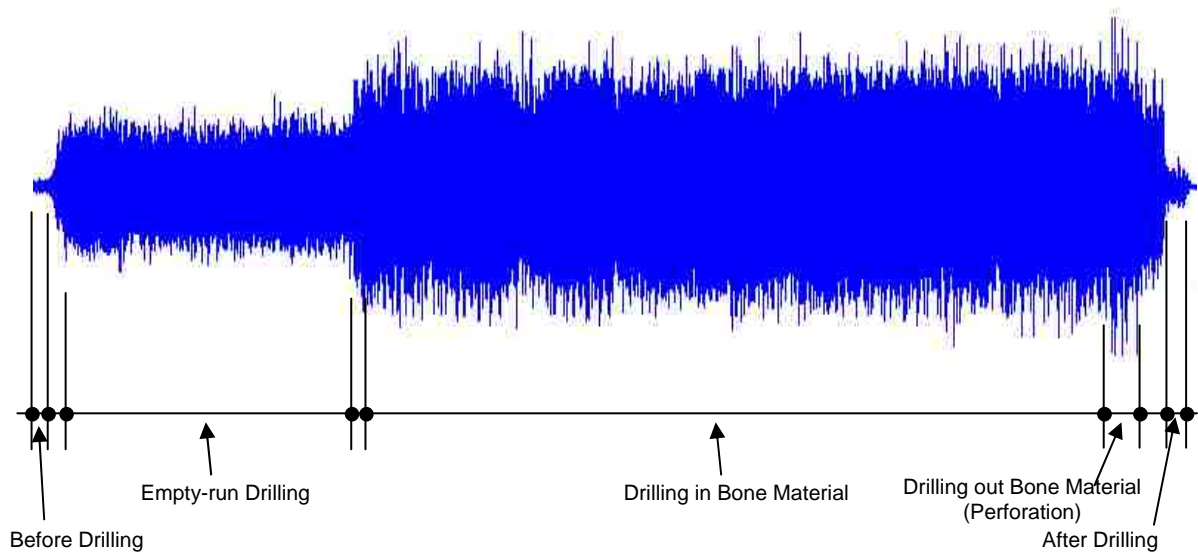


Figure 8.6. Different drilling states and transitions of drilling on cortical bone material in time domain

8.5.2. Drilling Sound Sources Analysis. For the hand-held power drills used in the orthopedic surgery, the sound sources are basically from the following categories:

1. Mechanical noises caused by solid structure vibrations and mechanical contact vibrations between different surfaces of mechanical components. The mechanical vibrations are transmitted through the machine structure to the

external surfaces which vibrate and radiate noise. The vibrations for hand-held drill are from unbalanced force of gear tooth meshes, shafts rotation, thrust bearings, fan blades, and commutator (bars and brushes), etc. Unbalance, misalignment, eccentricity are the main reasons for these vibrations. The vibrations may have same or different frequencies. The major noises of the mechanical vibration are from gear meshes and electric motor [Lang, 1999].

2. Cutting sound caused by the contact between the drill bit and the bone material. This kind of sound is from deformation and fracture of bone materials in the shear zone; deformation and fracture of drill bit between drill/bone materials. The cutting load can excite the drill bit to generate vibration sound. The vibrations of drill bit also affect some mechanical vibrations generated from the hand piece.
3. Aerodynamic noises caused by air disturbance. This kind of vibration from hand-held drill is mainly from the cooling fan with broad-band frequency at low shaft rotation speed. When the speed is high, aerodynamic forces are associated with the air volumes and flows from suction through to discharge and onwards down the piping system.
4. Electromagnetic noise caused by time-varying electromagnetic forces. This kind of noise only appears when AC voltage is applied to the motor.

Electromagnetic noise can be neglected because of the nature of the supply voltage if using AC power. For the drills with low rotation speed used in orthopedic surgery, aerodynamic noises can be neglected [Benko et al., 2004]. Thus, mechanical noises and cutting sounds need to be studied and analyzed. That means the final sounds the user can hear during bone drilling are the system sounds, including mechanical noises and cutting sound, which are caused by the tool-bone interaction and the drill hand-piece.

Power hand-held drills usually have two-stage planetary gear reduction sets driven by a DC motor. Therefore, the vibrations generated by the drill hand piece can be characterized by three shafts (motor shaft, transfer shaft, and output shaft), two sets of planetary gear meshes and an electric motor. Each of these elements influences the nature of the drill's vibration. For the two planetary gear sets, the following equations can be

used to calculate the gear reduction ratio and characteristic frequencies, such as the gear reduction ratio

$$R = \frac{\text{Input Rotation}}{\text{Output Rotation}} = 1 + \frac{T_r}{T_s} = \frac{f_{in}}{f_{out}} \quad (8.1)$$

where T_r is the number of teeth on the ring gear, T_s is the number of teeth on the sun gear, f_{in} is the input shaft frequency, and f_{out} is the output shaft frequency.

The spin frequency for the planet gears is

$$f_{spin} = \frac{T_r}{T_p} f_{out} \quad (8.2)$$

where T_p is the number of teeth on the planet gear.

The gear mesh frequency is simply the number of planet gear teeth multiplied by the planet spin frequency,

$$f_{mesh} = T_r f_{out} \quad (8.3)$$

Note that both the ring/planet and planet/sun meshes occur at the same frequency.

There are other remaining characteristic frequencies for the gears, such as the frequency at which an assembly phase passes through mesh and the frequency at which a specific tooth pair mates in mesh. However, invariably, f_{mesh} is the dominant component on any noise or vibration spectrum measured from a gearbox; also gear mesh is a much higher frequency than the shaft speed, the assembly phase passage frequency and the hunting tooth frequency [Lang, 1999]. In practice, both the mesh frequency carrier and the gear shaft frequency modulator can have many harmonics of various phasing present. This means that the resulting modulated signal can be more complicated with distinct tones appearing at frequencies of $n f_{mesh} \pm k f_{spin}$, where n and k are integer constants.

Normally, the terms symmetrically disposed about f_{mesh} (where n equals 1) will dominate. Sidebands associated with lower k value are usually the most detectable.

Similar to all the other power hand-held drills, the drill used in the experiments has two-stage gear reduction sets driven by a DC motor. Figure 8.7 shows the schematic illustration of the major components of the hand-held power drill used in the experiments. Using the equations 8.1, 8.2, and 8.3, the relationship can be obtained between the vibration frequencies and their sources for the gear sets of the power hand-held drill used as shown in Table 8.1.

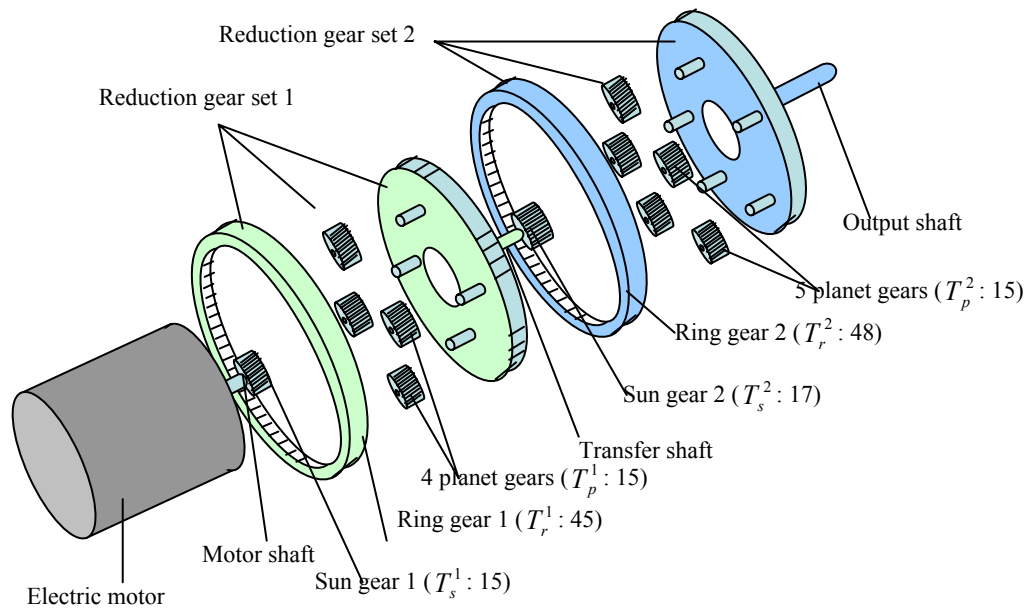


Figure 8.7. Schematic illustration of the major components of a hand-held power drill

For the vibrations from the electric motor, fault mechanism frequencies such as the contact rate between motor brushes and commutator segments, the magnetic pole-passage, are the major frequencies. The unbalanced magnetic forces can also cause

vibration. These frequencies are the harmonic frequencies of the motor shaft speed. That is

$$f_i = nif_{motor} \quad i = 1,2,3,\dots \quad (8.4)$$

where n is the number of potential impact points (motor brushes, commutator segments, or magnetic pole-passage), and i denotes higher harmonics.

Table 8.1. Relationship between the vibration frequencies and their sources

Items	Relative Speed	Operating Speed (Hz)
Shafts		1400RPM
Output	$f_{out}^1 = 1$	23.33
Transfer	$f_{out}^2 = 1 + \frac{T_r^1}{T_s^1} = 3.824$	89.23
Motor	$f_{motor} = (1 + \frac{T_r^1}{T_s^1})(1 + \frac{T_r^2}{T_s^2}) = 15.296$	356.91
Gear Meshes		
Secondary Mesh (Set 2)	$f_{mesh}^2 = T_r^2 f_{out}^1 = 48$	1119.84
Spin frequency (Set 2)	$f_{spin}^2 = \frac{T_r^2}{T_p^2} f_{out}^1 = 3.2$	74.66
Primary Mesh (Set 1)	$f_{mesh}^1 = T_r^1 f_{out}^2 = 172.08$	4014.63
Spin frequency (Set 1)	$f_{spin}^1 = \frac{T_r^1}{T_p^1} f_{out}^2 = 11.472$	267.64

8.5.3. STFT, PSD, Spectrogram and WT. The Fourier transform decomposed a signal or a function into a continuous spectrum of its frequency components, which when added together again recreates the original signal or function. In short words, this

transform separates the different frequencies along with their respective amplitudes, and makes it possible to detect and analyze the frequency content of any signal or function.

The general form is

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-i\omega t} dt \quad (8.5)$$

where t represents time; $x(t)$ is a continuous time function or signal; ω represents angular frequency: $\omega = 2\pi f$ (ω 's units are radians per second, f in hertz).

The inverse Fourier transform synthesizes a function from its spectrum of frequency components back to its time domain. The general form is

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega)e^{i\omega t} d\omega \quad (8.6)$$

FFT (Fast Fourier Transform) implements the discrete Fourier Transform with considerable savings in computational time. Fourier analysis makes implicit an assumption that the signal is stationary, i.e., it contains sinusoid components that are invariant for all time and the statistics parameters such as mean, root mean square do not have big change over time. However, most signals are non-stationary, so windowing (in a short time period) is used for this purpose. The other purposes of windowing technique are to decrease the variance of the power spectral density (PSD) estimate and to prevent leakage, which is the spreading of energy across the frequency spectrum caused by the FFT of the frequencies that are not periodic within the time interval of the signal [Vaseghi, 2005]. The short-time Fourier transform (STFT) is used to determine the sinusoidal frequency and phase content of the local sections of a signal as it changes over time. Simply stated, in the continuous-time case, the function to be transformed is multiplied by a window function that is nonzero for only a short period of time, i.e.,

$$X(\tau, \omega) = \int_{-\infty}^{\infty} x(t)w(t - \tau)e^{-i\omega t} dt \quad (8.7)$$

where $w(t)$ is the window function. $w(t)$ sets the value of $x(t)$ zero outside of the chosen interval and is used to avoid spectral leakage after doing Fourier Transform of a signal.

Common window functions that can be used in the STFT analysis are Rectangular, Triangular, Kaiser-Bessel, Hamming, Hann, and Blackman-Harris [Vaseghi, 2005]. Hann window is the most common window function used in signal processing and it is chosen in our analysis in order to reduce aliasing and leakage in Fourier transforms in addition to the short-time purpose. The Hann window function is shown below:

$$w[k+1] = 0.5\left(1 - \cos\left(2\pi\frac{k}{n-1}\right)\right) \quad k = 0, 1, \dots, n-1 \quad (8.8)$$

where n is the window size, a positive integer.

The length of the window determines the frequency resolution. The longer the window, the higher the frequency resolution. However, the sound signal of drilling in our research is not noise-free, thus using only one window would result in increased variance of the PSD estimate. To reduce variance, the signal need to be divided into several segments and the average of the periodograms of these segments is the used as the PSD estimates. To get a higher frequency resolution at low variance, 50% overlapping of the windows (segments) is used in this research.

Sound Pressure Level (SPL) or sound level L_p is a logarithmic measure of the rms (root mean square) sound pressure of a sound relative to a reference value:

$$L_p = 10 \log_{10}\left(\frac{p_{rms}^2}{p_0^2}\right) = 20 \log_{10}\left(\frac{p_{rms}}{p_0}\right) \quad dB \quad (8.9)$$

where p_0 is the reference sound pressure and p_{rms} is the rms sound pressure being measured.

To convert time-domain waveform into frequency domain sound level representation and get power spectrum, the following procedures need to be followed as shown in Figure 8.8.

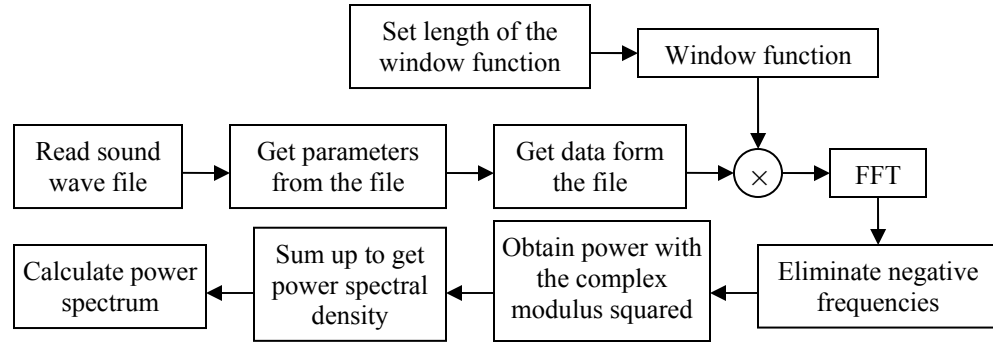


Figure 8.8. Block diagram of power spectrum calculation

Using STFT for the power spectrum in frequency domain does not have time information for a sound signal, so another way of represent the frequency spectra of windowed frames of a signal called spectrogram is used for later analysis and sound modeling. It uses STFT to plot a three-dimensional diagram of the energy of the frequency content of a signal as it changes over time. Usually, the horizontal and vertical axels represent time and frequency, respectively, and the color intensity of each point in the image represents the amplitude of a particular frequency at a particular time.

To calculate the spectrogram using the magnitude of the STFT is usually a digital process: sound sample data, in the time domain, is broken up into frames, which usually overlap, and Fourier transformed to calculate the magnitude of the frequency spectrum for each frame. Each frame then corresponds to a vertical line in the image; a measurement of magnitude versus frequency for a specific moment in time. Some spectrograms can be illustrated later in details.

Wavelet transform (WT) is also used in drilling sound characteristic analysis. WT has a good resolution in frequency and time domain synchronously, and it can extract more information in time domain at different frequency bands. The original signal is broken down into a series of local basis functions called wavelets. Each wavelet is located at a different position on the time axis and is local in the sense that is decays to zero when sufficiently from its center. Any particular local features of a signal can be identified from the scale and position of the wavelets into which it is decomposed. The

structure of a non-stationary signal can be analyzed into this way with local feature represented by close-packet wavelet of short length.

A wavelet is a waveform with effectively limited duration and zero average value. If given wavelet function is $\psi(t) \in L^2(R)$, wavelets are defined as

$$\psi_{a,b}(t) = \frac{1}{\sqrt{|a|}} \psi\left(\frac{t-b}{a}\right) \quad a, b \in R, a \neq 0 \quad (8.10)$$

where a is a scaling factor, b is a shifting factor, R is the set of real numbers, and $L^2(R)$ is the set of signals of finite energy.

The continuous wavelet transform (CWT) is defined as the sum over all time of the signal multiplied by scaled, shifted versions of the wavelet function,

$$[W_\psi x](a, b) = \frac{1}{\sqrt{|a|}} \int_{-\infty}^{\infty} \overline{\psi\left(\frac{t-b}{a}\right)} x(t) dt \quad (8.11)$$

where $x(t)$ is a continuous time function or signal, $\overline{\psi(t)}$ is complex conjugate function of $\psi(t)$.

Given a discrete sampling series $x(n)$ of a signal $x(t)$, the discrete wavelet transform (DWT) can be expressed by passing it through a series of filters. First the samples are passed through a low pass filter with impulse response g resulting in a convolution of the two:

$$y[n] = (x * g)[n] = \sum_{k=-\infty}^{\infty} x[k]g[n-k] \quad (8.12)$$

The signal is also decomposed simultaneously using a high-pass filter h . The outputs give the detail coefficients (from the high-pass filter) and approximation coefficients (from the low-pass). That means, the approximations are the high-scale, low-frequency components of the signal, and the details are the low-scale, high-frequency

components. It is important that the two filters are related to each other and they are known as a quadrature mirror filter. However, since half the frequencies of the signal have now been removed, half the samples can be discarded according to Nyquist's rule. The filter outputs are then down sampled by 2,

$$\begin{cases} y_{low}[n] = \sum_{k=-\infty}^{\infty} x[k]g[2n-k] \\ y_{high}[n] = \sum_{k=-\infty}^{\infty} x[k]h[2n-k] \end{cases} \quad (8.13)$$

Wavelet packets are particular linear combinations of wavelets. The key point of wavelet packet decomposition (WPD) is to extract information from the original signal by decomposing it into a series of approximations and details distributed over different frequency bands. The characteristics of frequency domain and time domain are preserved simultaneously. Figure 8.9 shows an example of multiple-level decomposition tree for wavelet transform. Wavelet packets form bases which retain many of the orthogonal, smooth and locate properties of their parent wavelets.

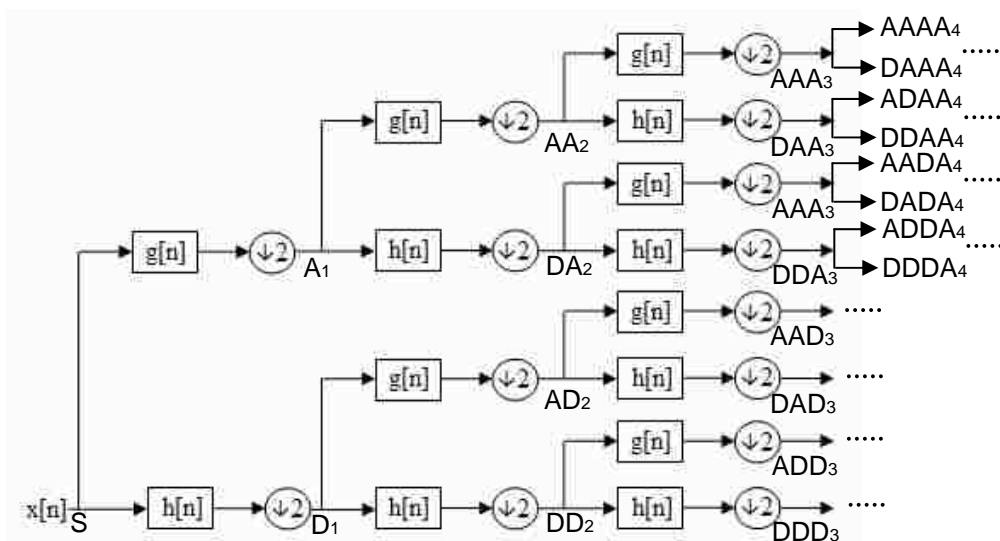


Figure 8.9. Multiple-level wavelet packet decomposition tree

Based on the relationship of frequency structure of wavelet decomposition, the frequency bandwidth of approximation and detail of level l are $\left[0, \frac{f_s}{2} 2^{-l}\right]$ and $\left[\frac{f_s}{2} 2^{-l}, \frac{f_s}{2} 2^{-l+1}\right]$ respectively, where f_s is the sample frequency of the original sound. It is noticed that the frequency band of every level is decomposed into two equal sub-bands, the approximation and the detail.

In this research, wavelet packet decomposition can be used for drilling sound characteristic analysis in addition to power spectrum analysis. Since the drilling sound sampling frequency f_s is 22.05 kHz for the sound analysis, the frequency bands of different decomposition levels for the original sound can be calculated using above mythology. For example, some of the decomposition frequencies of level 3 and level 4 shown in Figure 8.9 can be calculated as follows:

AAAA ₄ : [0, 689] Hz	DAAA ₄ : [689, 1378] Hz
ADAA ₄ : [1378, 2067] Hz	DDAA ₄ : [2067, 2756] Hz
AADA ₄ : [2756, 3445] Hz	DADA ₄ : [3445, 4134] Hz
ADDA ₄ : [4134, 4823] Hz	DDDA ₄ : [4823, 5512] Hz
.....	
AAD ₃ : [5512, 6890] Hz	DAD ₃ : [6890, 8268] Hz
ADD ₃ : [8268, 9646] Hz	DDD ₃ : [9646, 11025] Hz
.....	

Another way to use wavelet transform is to reconstruct signals using wavelet coefficients, therefore the signals for the wavelet packet decomposition components can be constructed and displayed in time domain. Some results using wavelet transform is given in Section 8.5.4.

8.5.4. Sound Analysis Using STFT and WT. In order to obtain drilling sound characteristics for different frequencies, sound analysis is also conducted in frequency domain. The captured sound files are initially down sampled to 22 kHz before spectral analysis. Power spectrum is used to obtain the characteristics of drilling sound, which can be represented by the relative amplitudes at different frequencies.

Free-running drilling sound is analyzed first using power spectral density (PSD) to extract the frequency features of the power hand-held drill during unloaded drilling as shown in Figure 8.10. This figure reveals that the major frequencies excited by the drill itself (drilling in the air) are more active in the frequency range of 400-6000 Hz, and the major amplitude peaks are around 5500 Hz, 4000 Hz, and 1000 Hz for the first, second, and third peak. To know exactly what these major peaks come from, detail sound source analysis for the dissembled power hand-held drill was conducted as shown in Section 8.5.2.

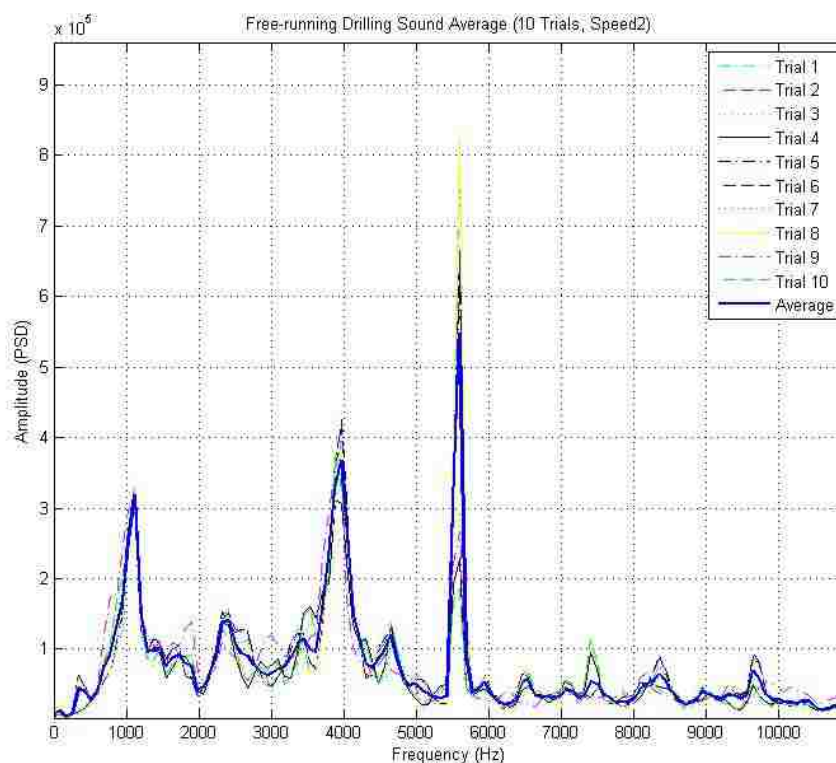


Figure 8.10. Power spectral density of free-running drilling sound

With detail analysis and calculation of the dissembled power hand-held drill as shown in Table 8.1, it is found that the 5500 Hz frequency is mainly from the vibration

frequencies of the third harmonic ($3\times$) of the magnetic pole-pass, $3\times$ brush/segment contact and motor fan's vibration, thus this frequency is primarily from the drill motor's vibration. It is also found that the 4000 Hz frequency is from the motor end planetary gear mesh vibration (reduction gear set 1) and the 1000 Hz frequency is from the output end gear mesh vibration (reduction gear set 2). The other big peaks such as 2300 Hz and 4600 Hz are from the harmonic frequencies of the output end gear mesh vibration. Some other peaks are from the harmonic frequencies of the vibration from the output shaft, the motor shaft, etc. Several frequencies of some peaks are not from the vibrations generated directly from the vibrations described above, therefore, a plausible explanation is that the gear mesh is experiencing some combination of both amplitude modulation and frequency modulation at the motor shaft speed [Lang, 1999].

It can be found that the frequencies of some peaks are in a small range from different trials. It is because small fluctuations in rotational speed imply smearing of power of speed-dependant components, thus instead of a single excited frequency component, a small frequency band is possible. It also can be found that the amplitudes of some significant frequencies are different for different trials. It is because sound sources of the drill are not ideally stationary but inherently comprise certain degree of unsteadiness, which results in fluctuating magnitude of the spectrum, and their magnitudes can differ by 300% or even more [Benko et al., 2004]. For example, the motor related vibration frequency (5500 Hz in this case) has different magnitudes for different trials.

Figure 8.11 shows the power spectral density of drilling sound from cortical bone material (106 pcf) bone drilling, cancellous bone material bone drilling (40 pcf, 30 pcf, 20 pcf). Some observations can be obtained:

1. For bone material drillings, all the drilling sound profiles in frequency domain are similar. Spectral analysis reveals that the frequencies of the major peaks in the experiment are still around 5500 Hz, 4000 Hz, and 1000 Hz for different kinds of bone material drilling. That means major spectral peaks have similar frequencies, which mainly come from the drill hand piece's mechanical vibration described above. The sound from the drill hand piece is the dominant noise and the cutting sound is relatively small.

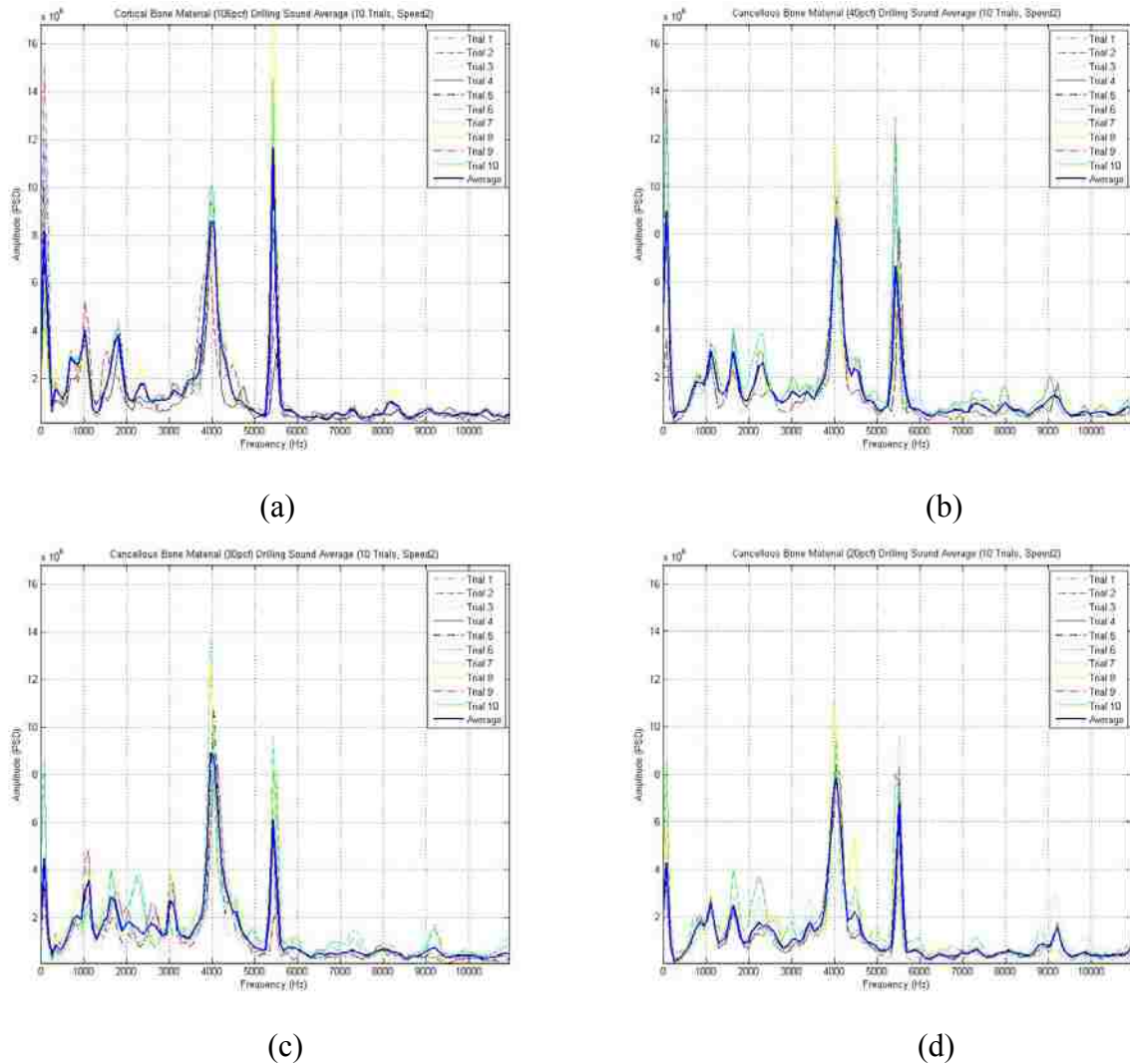


Figure 8.11. Power spectrum of bone drilling sound average: (a) cortical bone material drilling (106 pcf); (b) cancellous bone material drilling (40 pcf); (c) cancellous bone material drilling (30 pcf); (d) cancellous bone material drilling (20 pcf)

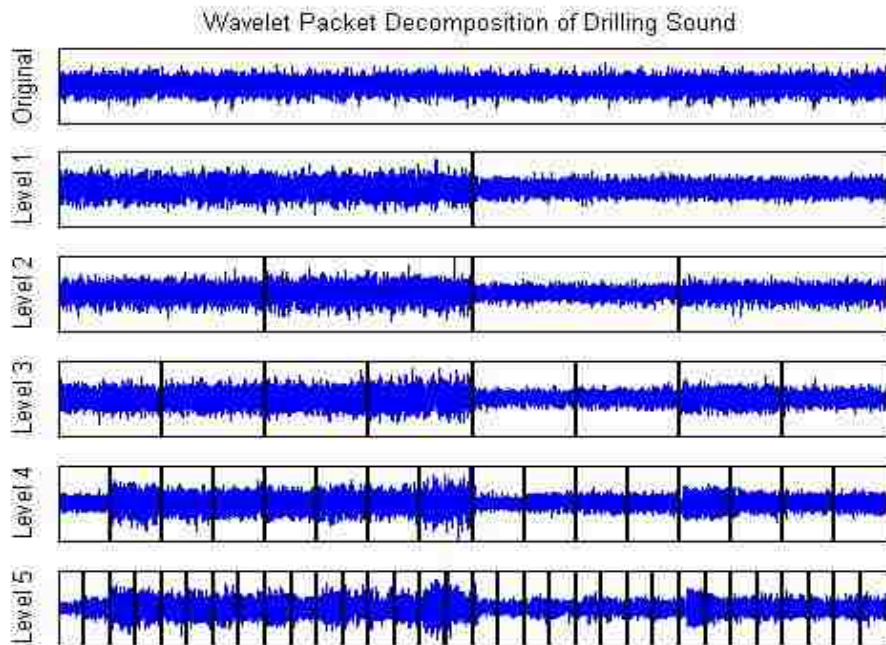
2. Spectral analysis reveals that the major peaks of the spectral amplitudes for all the drilling sounds are in the frequency range of 0-6000 Hz, and the variations of the spectral peaks from 6000-11000 Hz are not very significant. That means the overall system sound caused by drilling on different bone materials mainly happens in the frequency range of 0-6000 Hz. This is because the mechanical vibrations are usually concentrated on the first several harmonics of the vibrations [Basaraba and Archer, 1995].

3. From spectrum shown in Figure 8.11(a), it is shown that there is a frequency shift from 5500 Hz to 5400 Hz for free-running drilling and bone material drillings. This is mainly because drilling on the hard bone material will affect the output shaft's operating speed. This speed will affect the other mechanical vibration frequencies, e.g., gear meshes' frequencies, and thus the overall vibration frequencies will be lower. However, because the gear-set of the drill hand piece is a speed reduction system, the three shafts have different rotation speeds. A small speed variation of the output shaft will create a large variation on the motor side, but a relative small variation on the transfer shaft. Thus, the motor shaft related frequencies are changed a lot and the frequency shift is very obvious.
4. Bone density influences the drilling sound. The harder the bone material, the bigger the amplitude of drilling sound by comparing the cortical bone material and cancellous bone materials, especially at the frequency range of 0-6000 Hz. As the drill makes contact the bone, the harder bone will need more force to drill through, therefore the cutting sound is increased. Furthermore, it will increase the levels of unbalance, misalignment and eccentricity of the shafts, gears, and motor armature. Therefore, the overall unbalanced force is increased and the amplitude of the sound is increased. Still, the hand-piece sound is the dominant sound.
5. The drilling sound of the cortical bone material versus that of the cancellous bone material can be identified from Figure 8.11 (a) and Figure 8.11 (b)-(d). The first spectral peak of cortical bone drilling appears at around 5400 Hz, followed by 4000 Hz. But the first spectral peak of the cancellous bone drilling appears at 4000 Hz, and the second is around 5400 Hz. That means there is a frequency shift for different kinds of bones. This is because different loads influence the vibrations of the gears (4000 Hz) and the motor (5400 Hz), and give different results. For example, when the loads decrease, both unbalanced forces from the gear mesh and the motor are decreased, but the motor related unbalanced forces drop quicker than those of gears because of

the motor's mechanical structures. It is easy to identify different bone materials using these noticeable “spikes”.

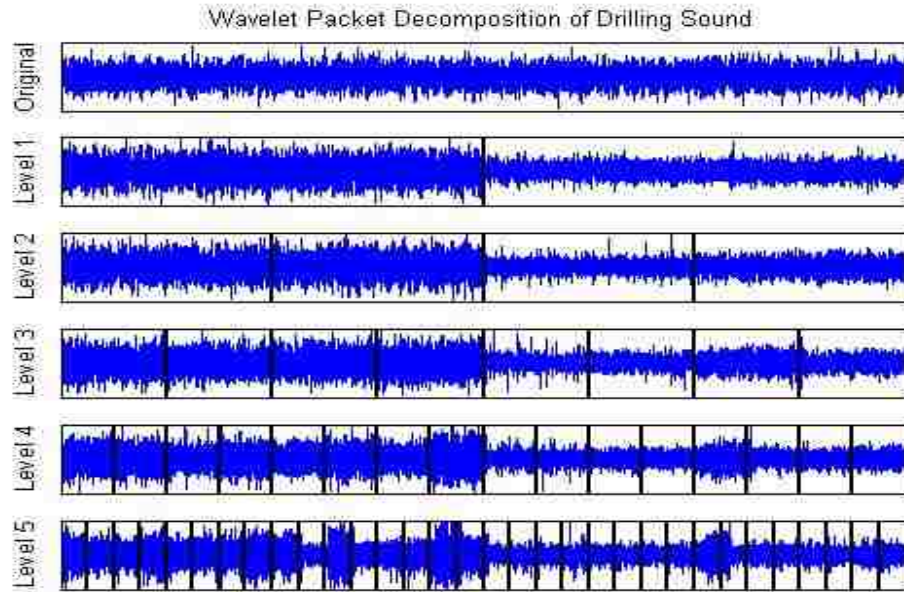
6. The difference of drilling amplitudes among cancellous bones is not very significant because the material density range is very small (20-40 pcf) as shown in Figure 8.11 (b)-(d). Therefore, any of these bone materials can be used for cancellous bone drilling.

Figure 8.12 shows the examples of wavelet packet decompositions of (a) the free-running drilling sound, (b) the cortical bone material drilling sound and (c) the cancellous bone material drilling sound.

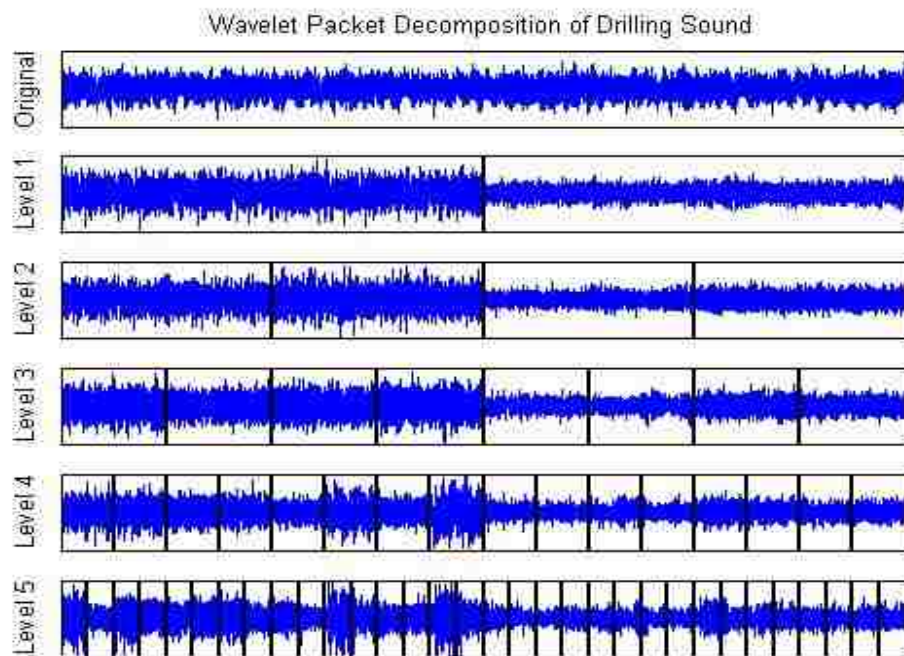


(a)

Figure 8.12. Wavelet packet decomposition of different drilling sounds: (a) free-running drilling sound; (b) Cortical bone material drilling sound; (c) cancellous bone material drilling sound



(b)



(c)

Figure 8.12. Wavelet packet decomposition of different drilling sounds: (a) free-running drilling sound; (b) Cortical bone material drilling sound; (c) cancellous bone material drilling sound (cont.)

From above figures, the same drilling sound characteristics can be obtained as those of using power spectrum. For example, in level 1 decomposition, the signal energy in D1 is much smaller than that in A1, therefore the drilling sound is more active in the frequency range of 0-6000 Hz; the frequency bands of 0-344, 5167-5512 Hz and 3790-4131 Hz of decomposition at level 5 carries more energy for the cortical bone and cancellous bone material drilling. Wavelet packet decomposition can supply frequency resolution using frequency bandwidth; therefore it is easy to identify the sound features using different frequency ranges. However, it is very hard to know the exactly frequency values for each major peaks and it is hard to map the sound energy to the human ear's listening capability.

Figure 8.13 shows a spectrogram of drilling sound on the 3rd generation composite humerus. Different states of drilling sound can be classified similar to the drilling on cortical bone material: free-running/idle drilling, drilling into bone, drilling in bone (cortical bone), drilling in bone (cancellous bone), drilling in bone (cortical bone), drilling out bone (perforation), and transitions from the above states in sequence. As shown, different colors represent the amplitudes of the drilling sound (warm colors represent higher dB in sound intensity and cold colors represent lower dB), and time and frequency information are represented by the horizontal and vertical axels respectively.

The following observations can be made from Figure 8.13:

1. The spectrogram can show different drilling states and transitions by observing the color changes in the image. There are several noticeable vertical lines (representing by color changes) which can separate the different drilling states.
2. For the “before drilling” and “after drilling” states, the colors are cold. That is because these two states only have environment noise. Therefore the sound intensity is very low.
3. For the “free-running” state, the colors are much “warmer” than those of the “before drilling” state. The red line around 5500 Hz represents the dominate frequency which has the biggest amplitude/intensity. The frequencies around 4000 Hz, 1800 Hz and 1100 Hz have higher intensities, and the colors of these

frequencies are close to red. The frequency range from 0-400 Hz has a kind of “green-yellow” color which represents low intensity.

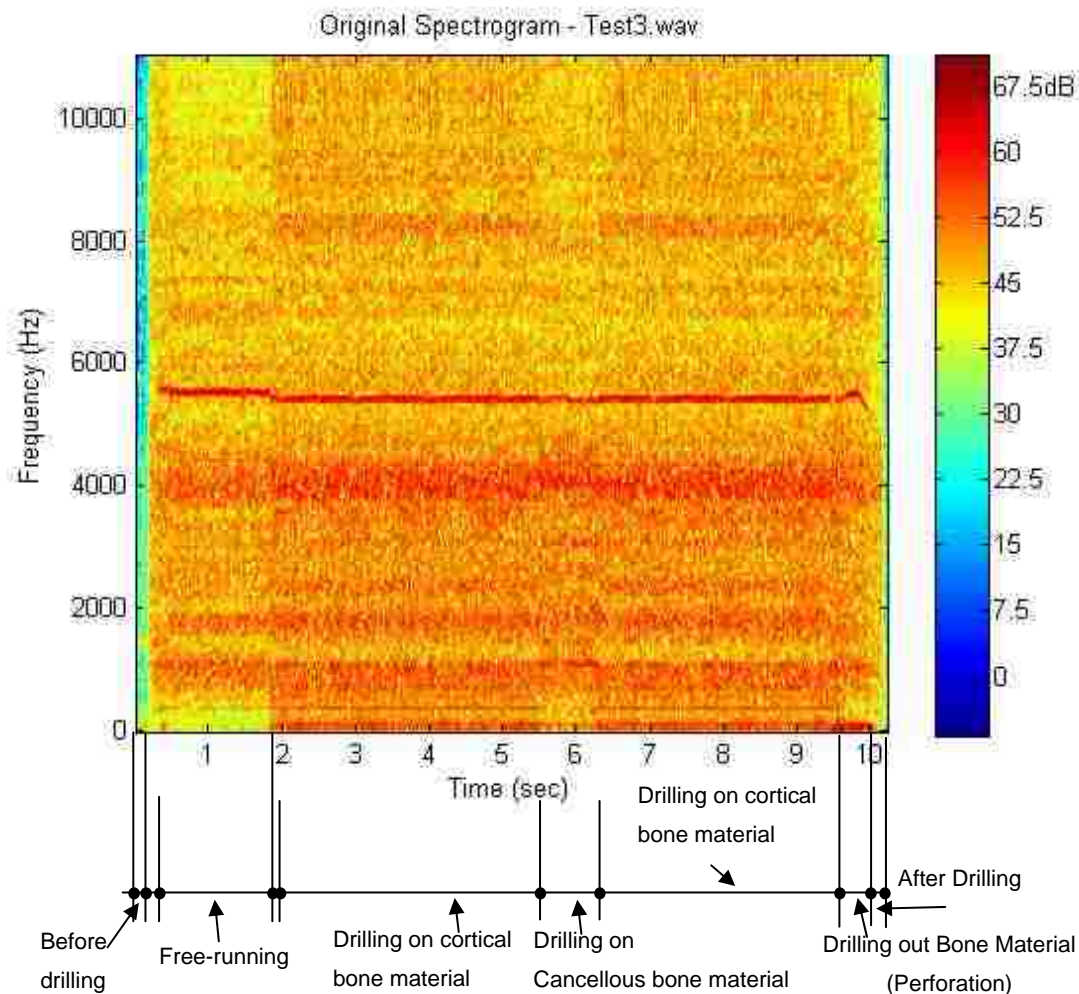


Figure 8.13. Spectrogram of drilling sound for the 3rd generation composite humerus

4. For the “drilling on cortical bone” states, it is obvious that the overall colors are much “warmer” than those of the “free-running” state, because the overall sound intensity increases. There is a frequency shift from 5500 Hz to 5400 Hz

since drilling on the bone material will affect the output shaft's operating speed. The frequency range from 0-400 Hz has a kind of "yellow-red" colors which represent higher sound intensity compared to those of the "free running" states.

5. For the "drilling on cancellous bone" state, the overall colors are between those of the "free-running" state and those of the "drilling on cortical bone" state, especially at the low frequency range of 0-400 Hz.
6. For the "drilling out bone material (perforation)" state, there is a noticeable frequency change around 5500 Hz. Also amplitudes of some frequencies are very big, e.g., around 200 Hz. This is because "perforation" carries biggest energy.

The signals can be constructed using wavelet coefficients for each wavelet packet decomposition components and display them in time domain. The reconstructed signal can be used to analyze different drilling states and transitions. By comparing with different reconstructed components in time domain, it is easy to know some additional information which spectrogram can not easily supply.

Figure 8.14 shows an example of the drilling sound decomposed as different components in different time windows and different frequency bands. Since high frequency range of 5512-11025 Hz is not very active, the figure only shows the wavelet decompositions down to level 4 in the frequency range of 0-5512 Hz and level 3 in the range of 5512-11025 Hz.

This figure can supply some easier observations for the different drilling states and transitions. For example, "before drill running" and "after drilling" states can be found in almost all the plots; the transitions from "free-running" to "cortical bone drilling"(around 1.8 sec), "cortical bone drilling" to "cancellous bone drilling" and back to "cortical bone drilling" (around 6 sec) can be seen clearly from the signal energy changes shown in AAAA₄ and AAD₃. The energy burst of "perforation" can be seen from AAAA₄ and DAAA₄, in which the frequency bands are [0, 689] Hz and [689, 1378] Hz.

Figure 8.15 shows some detail information for two frequency bands. From this figure, it can be easily found the transition for the different drilling states. For example,

the red arrows shown in the figure can provide us more detail information which some other frequency bands can not give.

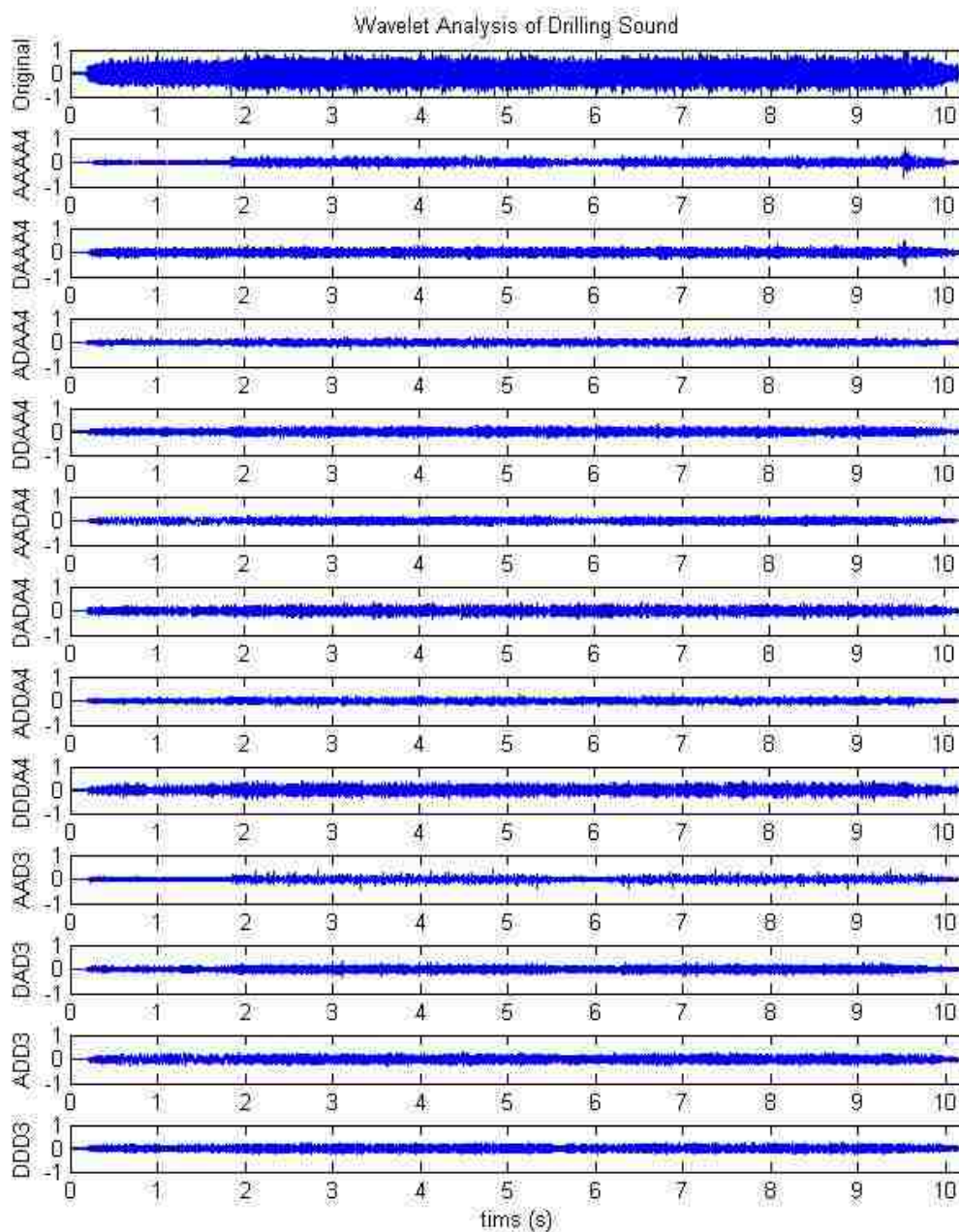


Figure 8.14. Wavelet analysis of drill sound on the 3rd generation composite humerus

Figure 8.16 shows the displacement of drill-bit vs the wavelet analysis results for the frequency bands of $[0, 689]$ Hz and $[5512, 6890]$ Hz. It is very clear that the different drilling states have different drilling sounds. Therefore, the characteristic results of Figure 8.15 and Figure 8.16 can be used to identify different drilling states and apply them to the drilling model.

From the above analysis and comparisons for STFT and WT, it can be found that each of them has its own advantages and disadvantages. STFT is better used in the frequency domain analysis and the values of the frequencies can be obtained. WT is better used in the time domain analysis with bandwidth of frequency ranges. However, to analysis the drilling states and transitions, both of them can be used to get better results.

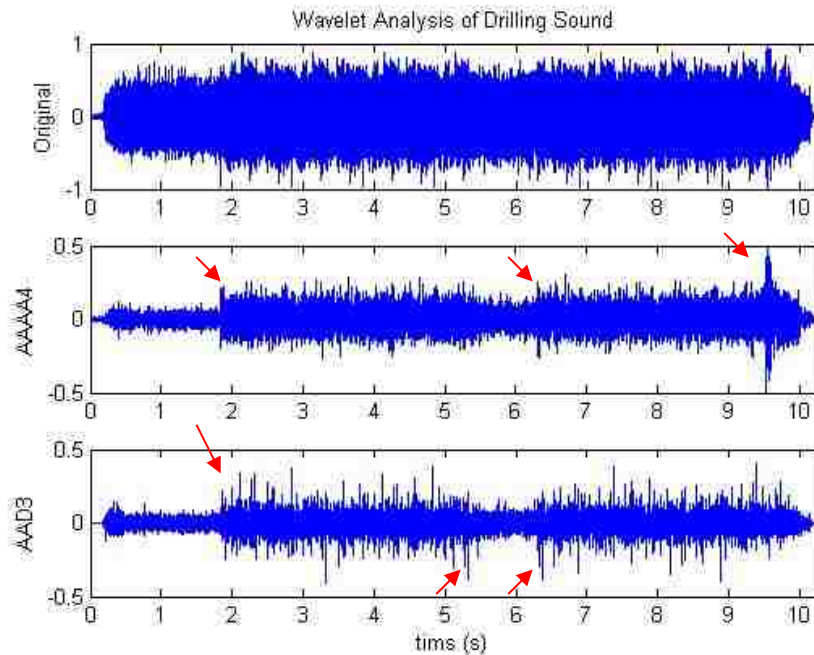


Figure 8.15. Wavelet analysis for the frequency bands of $[0, 689]$ Hz and $[5512, 6890]$ Hz

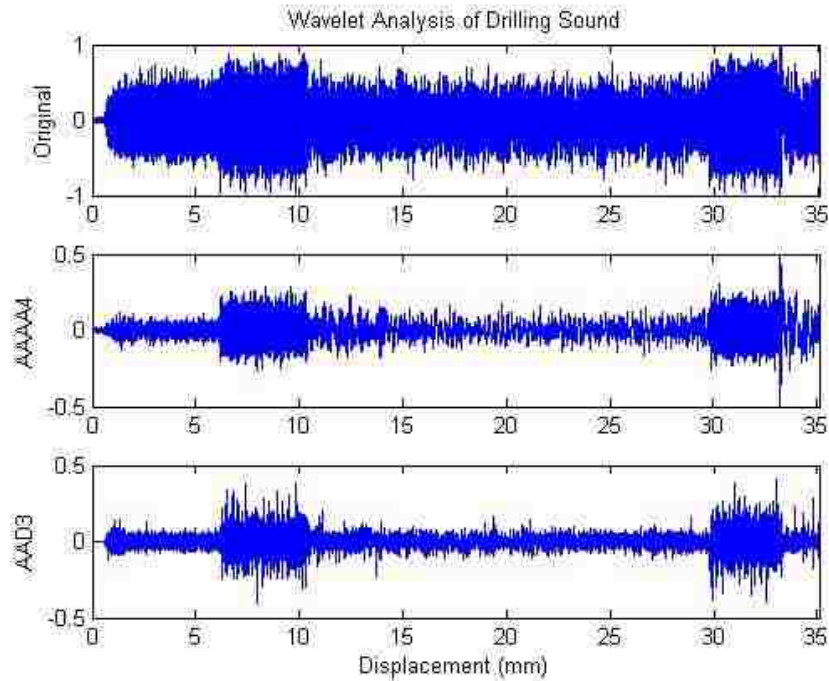


Figure 8.16. Displacement vs. wavelet analysis for the frequency bands of $[0, 689]$ Hz and $[5512, 6890]$ Hz

8.5.5. Spectral Subtraction. Spectral subtraction [Boll, 1979] is a very simple and effective method usually used for noise reduction. In this research, this method is applied to obtain the drilling sound difference caused by tool-bone interaction. Since the final sounds the users hear during bone drilling are the system sounds, sound difference is from the mechanical noise changes and the cutting sound. The sound difference can be used for sound characteristic analysis as well.

To investigate sound difference, an assumption can be made that the free-running sound generated by the drill hand-piece is a wide-band, stationary noise. Therefore, in this method, the average bone drilling spectrum and the average free-running sound spectrum are estimated and subtracted from each other, and then the sound difference can be obtained. The block diagram of spectral subtraction is given in Figure 8.17. Following this diagram, the sound difference caused by tool-bone interaction can be obtained in time domain after inverse fast Fourier transform (IFFT).

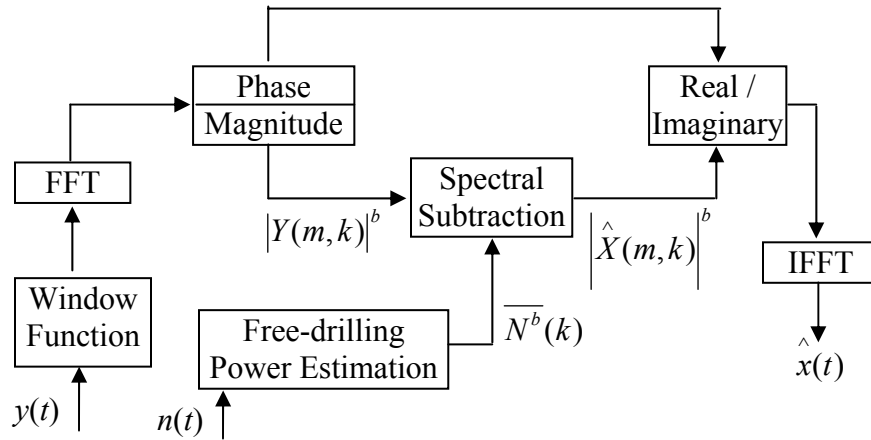


Figure 8.17. Block diagram of spectral subtraction

In time domain, the bone drilling sound signal $y(t)$ is composed of the sound difference $x(t)$ caused by tool-bone interaction and the drill free-running sound $n(t)$ as

$$y(t) = x(t) + n(t) \quad (8.14)$$

After windowing the signal and taking Fourier transform of both sides, the equation is

$$Y_w(e^{j\omega}) = X_w(e^{j\omega}) + N_w(e^{j\omega}) \quad (8.15)$$

where $Y_w(e^{j\omega})$, $X_w(e^{j\omega})$, and $N_w(e^{j\omega})$ are the Fourier transforms of the bone drilling sound signal, the sound difference caused by tool-bone interaction and the drill free-running signal $y(t)$, $x(t)$, and $n(t)$ respectively. To simplify the notation the w subscript is dropped. Multiplying both sides by their complex conjugates leads to

$$|Y(e^{j\omega})|^2 = |X(e^{j\omega})|^2 + |N(e^{j\omega})|^2 + 2|X(e^{j\omega})||N(e^{j\omega})|\cos(Dq) \quad (8.15)$$

where Dq is the phase difference between the two signals. Taking the expected value of both sides:

$$\begin{aligned} E\{|Y(e^{j\omega})|^2\} &= E\{|X(e^{j\omega})|^2\} + E\{|N(e^{j\omega})|^2\} + E\{2|X(e^{j\omega})||N(e^{j\omega})|\cos(Dq)\} \\ &= E\{|X(e^{j\omega})|^2\} + E\{|N(e^{j\omega})|^2\} + 2E\{|X(e^{j\omega})|\}E\{|N(e^{j\omega})|\}E\{\cos(Dq)\} \end{aligned} \quad (8.17)$$

Thus, the noise can be subtracted from the mixed signals as

$$\left|\hat{X}(e^{j\omega})\right|^b = |Y(e^{j\omega})|^b - |N(e^{j\omega})|^b \quad (8.18)$$

where $\left|\hat{X}(e^{j\omega})\right|^b$ is an estimate spectrum of the sound difference, $|Y(e^{j\omega})|^b$ is the bone drilling sound spectrum, and $|N(e^{j\omega})|^b$ is the drill free-running noise spectrum. The exponent b is set to 1 when magnitude spectral subtraction is performed ($E\{\cos(Dq)\} = 1$), and set to 2 in the case of power spectral subtraction ($E\{\cos(Dq)\} = 0$).

Spectral subtraction may be implemented in the power or the magnitude spectral domains. However, because phase difference between the signals can be neglected in this research, i.e., $E\{\cos(Dq)\} = 0$, short time power spectral subtraction is applied. $Y(m, k)$ is the short-time spectrum of the drilling sound signal $y(n)$ for each frequency bin k and each frame m , where it can be calculated by windowing the signal $y(n)$ with Hann window, and then transforming via discrete fast Fourier transform. $N^2(k)$ is the time-average of estimated spectrum of the drill free-running noise signal at bin k . The power spectrum of estimated signal $\left|\hat{X}(m, k)\right|$ is calculated as

$$\left| \hat{X}(m, k) \right|^2 = \begin{cases} |Y(m, k)|^2 - N^2(k), & \text{if } |Y(m, k)|^2 - N^2(k) > 0 \\ 0, & \text{otherwise} \end{cases} \quad (8.19)$$

Since human can not detect the phase distortion of audio signal, the phase function of sound difference $\theta_y(m, k)$ is combined with the estimated spectrum $\hat{X}(m, k)$ and inversely transformed from the frequency domain into time domain by IFFT and overlap added method:

$$\hat{x}(t) = IFFT \left\{ \left| \hat{X}(m, k) \right| e^{j\theta_y(m, k)} \right\} \quad (8.20)$$

Therefore, the sound difference caused by tool-bone interaction can be estimated using $\hat{x}(t)$ in time domain.

Figure 8.18 shows the comparison of bone drilling sound of drilling on cortical bone material, drill free-running sound and sound difference in frequency domain. Similar observations can be found between Figure 8.18 and Figure 8.12(a), such as major peaks, dominant frequencies, etc. From Figure 8.18, it is more obvious that the drilling sound happens in the frequency range of 0-6000 Hz, and the variations of the spectral peaks from 6000-11000 Hz are not very significant. That means from spectral reduction, in the frequency range of 0-6000 Hz, the spectrum of the drilling sound difference is almost same as that of the bone drilling sound sound, especially all the major peaks; in the higher frequency range of 6000-11000 Hz, the spectrum of the drilling sound difference is between that of the bone drilling sound and that of the free-running sound. The frequency shift at 5500 Hz to 5400 Hz is also very apparent. Figure 8.19 shows the result of the drilling sound difference in time domain.

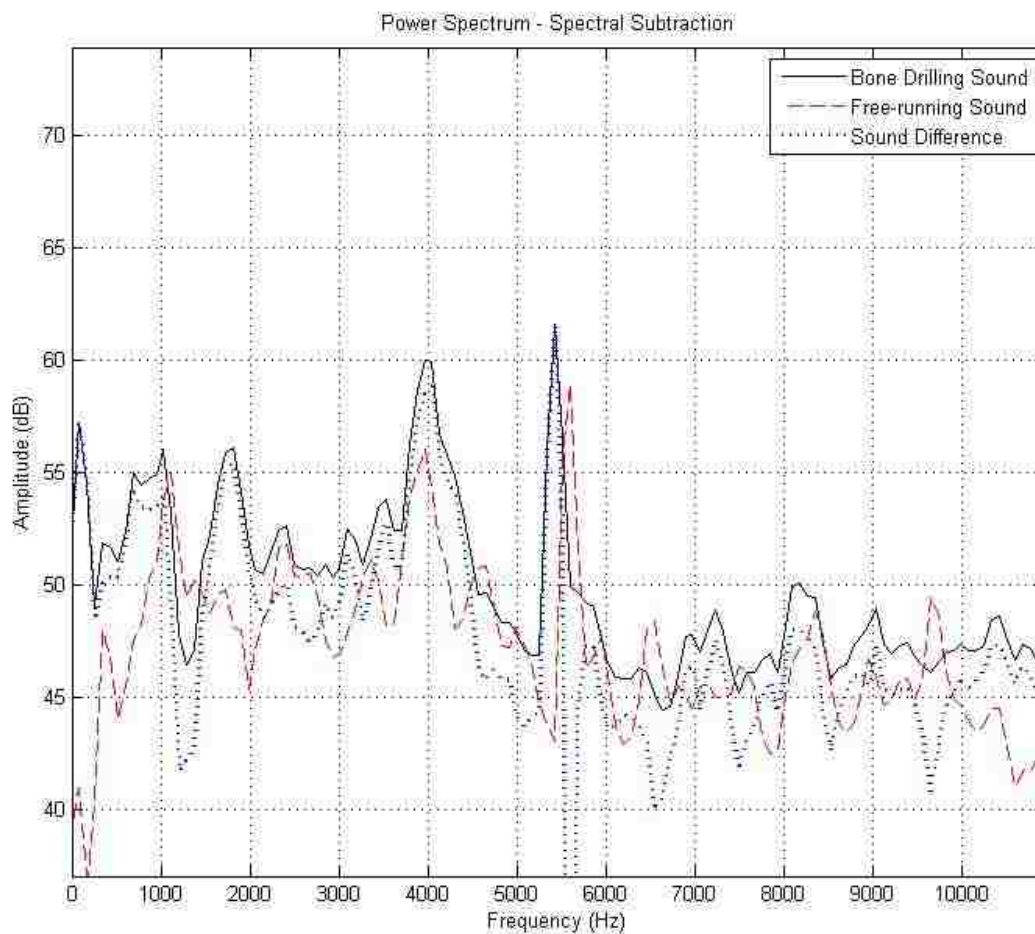


Figure 8.18. Spectral subtraction result: comparison of bone drilling sound, drill free-running sound and drilling sound difference in frequency domain

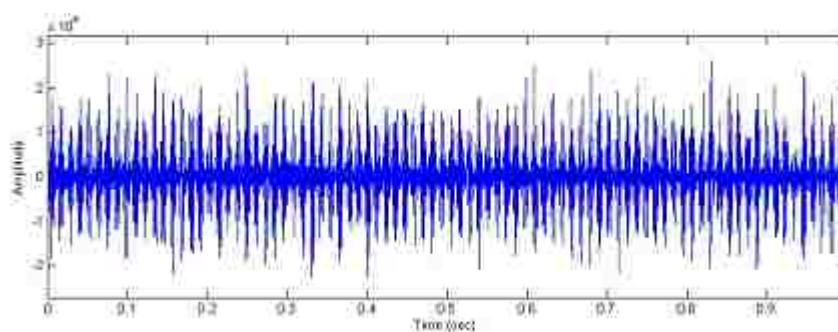


Figure 8.19. Spectral subtraction result: drilling sound difference in time domain

8.6. SOUND MODELING

Based on the knowledge on drilling sound characteristics and the qualitative and quantitative relations obtained, sound models are developed and used to generate the synthetic sound in virtual bone surgery.

A sound model assumes certain sound waveform or sound generation mechanism characteristics. Every sound analysis/synthesis system has an underlying model. The sounds produced by a musical instrument, or by any physical system, can be modeled as the sum of a set of sinusoids plus a noise residual. Due to the complexity of machining sound in real bone surgery, a spectral sound modeling method, Spectral Modeling Synthesis (SMS), is presented for the virtual bone surgery simulation to find the sinusoids and residual. SMS is used to find the mathematical model of the free-drilling sound, and sound characteristic is applied to this model to obtain the other kinds of drilling sound, such as cortical bone drilling, cancellous bone drilling, etc.

The general form of SMS can be written as [Serra, 1989]:

$$s(t) \approx \hat{s}(t) = \sum_{k=1}^K A_k \sin(\omega_k t + \theta_k) + r(t) \quad (8.21)$$

where $s(t)$ is the input sound, input sound, A_k , ω_k and θ_k are the instantaneous amplitude, frequency and phase of the k th sinusoid, and $r(t)$ is the residue.

The sinusoidal, or deterministic, component normally corresponds to the main modes of vibration of the system. A deterministic signal is traditionally defined as anything that is not noise (i.e., an analytic signal, or perfectly predictable part, predictable from measurements over any continuous interval). However, in this discussion the class of deterministic signals considered is restricted to sums of quasi-sinusoidal components (sinusoids with slowly varying amplitude and frequency). Each sinusoid models a narrowband component of the original sound and is described by an amplitude and a frequency function.

The residual comprises the energy produced by the excitation mechanism which is not transformed by the system into stationary vibrations, plus any other energy component that is not sinusoidal in nature. The residual, a stochastic signal is fully

described by its power spectral density, which gives the expected signal power versus frequency. When a signal is assumed to be stochastic, it is not necessary to preserve either the instantaneous phase or the exact magnitude details of individual FFT frames.

Assuming that $r(t)$ is a stochastic signal, it can be described as filtered white noise,

$$r(t) = \int_0^t h(t, \tau) u(\tau) d\tau \quad (8.22)$$

where $u(t)$ is white noise and $h(t, \tau)$ is the response of a time-varying filter to an impulse at time t . The residual is modeled by the convolution of white noise with a time-varying frequency-shaping filter. $r(t)$ can also be expressed as

$$r(t) = s(t) - \sum_{k=1}^K A_k \sin(\omega_k t + \theta_k) \quad (8.23)$$

Figure 8.20 shows the block diagram of the analysis for the Sinusoidal plus Residual Model. Sinusoidal Modeling is first conducted to get sine frequencies, magnitudes, and phases. After performing the STFT for each windowed portion of the signal, a series of complex spectra are computed, from which the magnitude spectra is calculated. From each spectrum the prominent peaks are detected and the peak trajectories are obtained utilizing a peak continuation algorithm. Residual modeling is used to obtain the stochastic component by subtracting the deterministic components from the signal in time domain. After using the same windowing parameters, the envelopes of the spectra are then approximated using a line-segment approximation. These envelopes form the stochastic representation.

8.6.1. Sinusoidal Modeling. Sinusoidal modeling assumes that each spectrum of the STFT representation can be explained by a series of sinusoids. For a given frequency resolution, with enough points in the spectrum, a sinusoid can be identified by its shape [Serra, 1989]. Because most natural sounds are not perfectly periodic and do not have nicely spaced and clearly defined peaks in their frequency domains as in the bone

drilling sound in this research, it is impossible to directly obtain sinusoidal components. Practical solutions must be developed to find the sinusoidal component of the input sound.

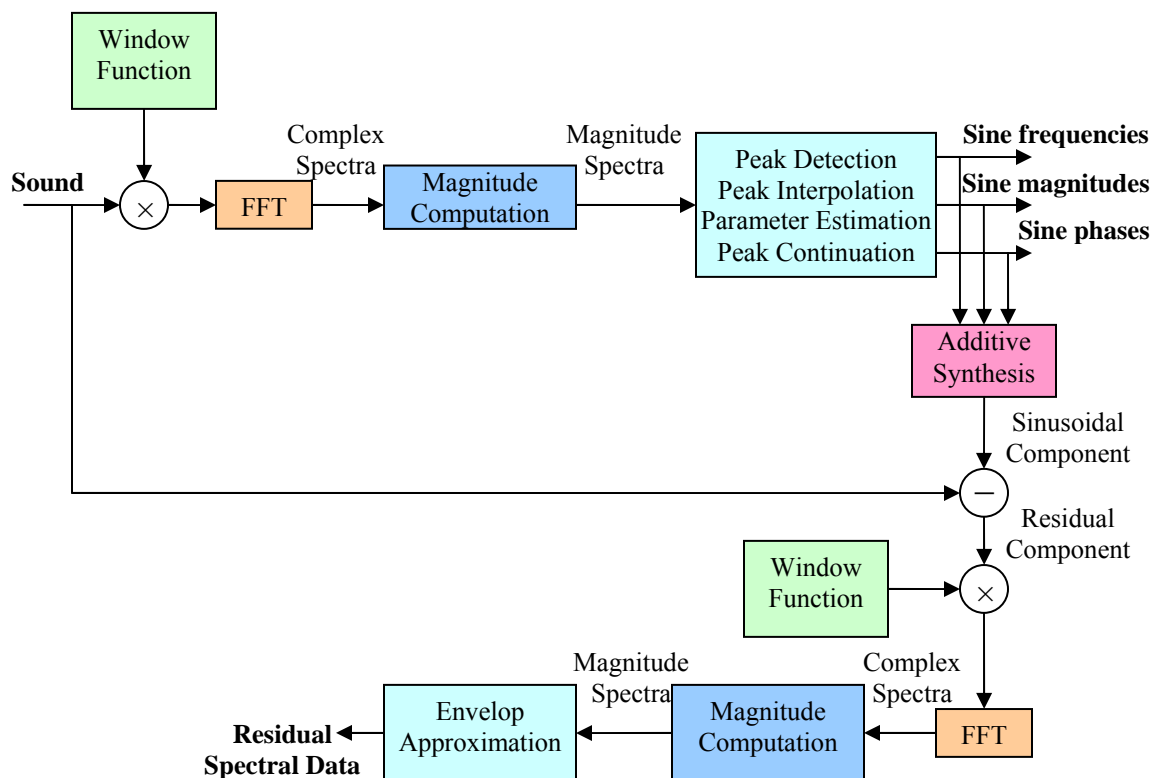


Figure 8.20. Block diagram of spectral modeling synthesis

The following steps are very important for sinusoidal modeling: **Peak Detection** → **Peak Interpolation** → **Parameter Estimation** → **Peak Continuation**. First, meaningful peaks in the input signal are detected. Second, the peaks are interpolated to obtain better frequency resolution. Third, the amplitudes and phases of the detected peaks are estimated. Finally the peaks are connected into trajectories.

Peak detection is a crucial part of a sinusoidal modeling system, since sinusoidal synthesis is performed using the detected peaks only. A peak is defined as a local maximum in the magnitude spectrum, and the only practical constraints for the peak search are frequency range and a magnitude threshold. After input sound is read and transformed into the frequency domain using the FFT and processed separately, the maximum and average magnitudes of the spectral frame are computed and stored. The following steps are then repeated until either a specified maximum number of peaks have been located or no more peaks are available:

1. The maximum-magnitude bin in the frame is located within the specified frequency range;
2. If the ratio of its magnitude to the average magnitude of the frame is below a specific threshold, it is assumed to be noise and it is deduced that no more peaks are present;
3. If its magnitude is above a specified absolute threshold, it is added as a sinusoidal peak and the bins it covered are zeroed out in the analysis frame.

All the sinusoidal peaks and FFT frames can also be pre-computed and stored. All peaks in a frame are found by locating bins where the derivative of the magnitude changes from positive to negative. The peaks for each frame are stored in the order of decreasing magnitude. At run-time the predefined maximum numbers of peaks that satisfy the frequency and threshold bounds are selected per frame for peak interpolation.

The efficient spectral interpolation scheme presented by Serra [1989] is applied here to refine the estimation accuracy. Considering the sampled nature of the spectrum returned by the STFT, each peak is accurate only to within half a sample. A spectral sample represents a frequency interval of f_s/N Hz where f_s is the sampling rate and N is the FFT size. Thus, for high quality sampling frequencies, more samples are required. To obtain high frequency accuracy, a quadratic function, e.g., parabolic interpolation, can utilize only three samples immediately surrounding the maximum magnitude for accurate frequencies of the sinusoidal components. The frequency and magnitude of a peak can be obtained from the magnitude spectrum expressed in dB. Then the phase value of the peak can be measured by reading the value of the unwrapped phase spectrum at the position

resulting from the frequency of the peak. Figure 8.21 shows the temporary results of peak detection on the spectrums of the drill free-running sound.

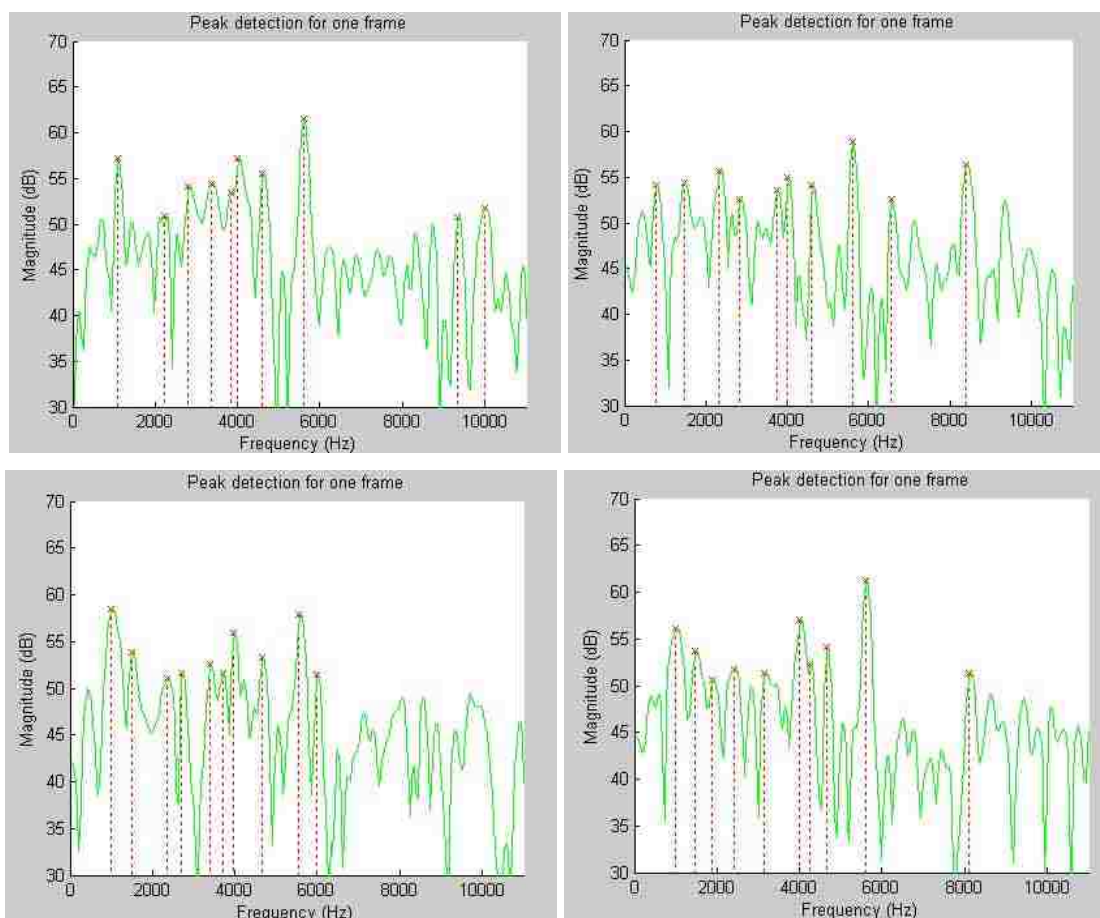


Figure 8.21. Temporary results of peak detection for four different frames of free-running sound

Even using the most advanced methods, it is difficult to estimate the sinusoidal parameters of a complex sound by analyzing the signal only once in each time frame. One possibility of the analysis to estimate the parameters with a simple estimation method, and then iteratively improve the parameter set [Depalle and Helie, 1997]. Starting from the estimated values of sine frequencies, magnitudes, and phases which are

obtained from peak detection and peak interpolation, it is easy to iteratively improve accuracy of the estimates. First, the amplitudes and phases are solved, assuming that the frequencies are correct. Then, the frequency estimates are improved, assuming that the amplitudes and phases are correct. This procedure is repeated several times, resulting in better estimates for the parameters at each iteration. During the iteration process, the number of sinusoids can be altered, so it is necessary to remove and add sinusoids.

The estimated sine frequencies, magnitudes, and phases of the prominent peaks in a given frame are sorted by frequency. Once these spectral peaks have been detected, a subset of them is organized by the peak continuation algorithm into peak trajectories, with each trajectory representing a stable sinusoid. The design of such an algorithm can be approached as a line detection problem where, out of a surface of discrete points (each one being a peak), the algorithm finds lines according to the characteristics imposed by the model.

The sinusoidal model assumes that each of these peaks is part of a frequency trajectory and the peak continuation algorithm is responsible for assigning each peak to a given “track”. The original method presented by McAulay and Quatieri [1986] in their sinusoidal representation is based on finding, for each peak, the closest one in frequency in the following frame, as shown in Figure 8.22.

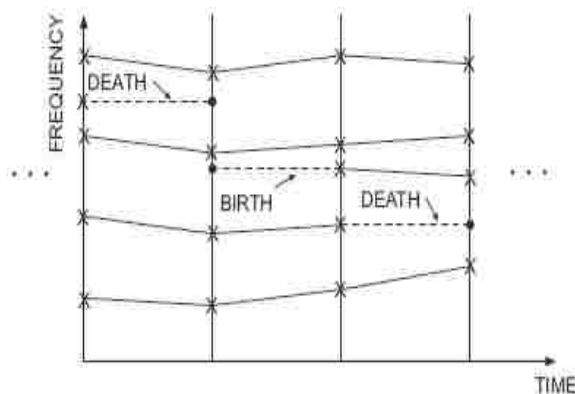


Figure 8.22. Original peak continuation algorithm [McAulay and Quatieri, 1986]

8.6.2. Residual Modeling. Once the sound's deterministic component has been detected, the next step is to obtain the residue, which in a simplified form becomes the stochastic component. The subtraction can be performed in time domain or frequency domain, depending on whether the deterministic component preserves the phases of the original sound or not. In this research, the subtraction is done in time domain because the phases are preserved. The analysis of the spectral residue is performed in frequency domain with same analysis window, window length, FFT size, and hop size for STFTs.

An assumption underlying the Sinusoidal plus Residual model is that the residual signal is quasi-stochastic. This implies that the residue is fully described by its amplitude and its general frequency characteristics. Each magnitude-spectrum residual can be approximated by its envelope, since only its shape contributes to the sound characteristics. This type of problem is generally solved by performing curve fitting [Strawn, 1980], i.e., finding a function that matches the general contour of a given curve, which in this research is a magnitude spectrum. Standard techniques are spline interpolation, the method of least squares, or the straight-line approximations [Serra and Smith, 1990].

A line-segment approximation is accurate enough and gives the desired flexibility. It is performed by stepping through the magnitude spectrum and finding local maxima in every defined section. The resulting points are equally spaced and are connected by straight lines to create the spectral envelope. The accuracy of the fit is given by the hop size, which is set depending on the sound complexity.

Compared to the sinusoidal analysis and synthesis, the processing of the stochastic part is significantly simpler. Basically the only parameters of the stochastic analysis that can be adjusted are the window length and frame rate.

8.6.3. Sound Synthesis. Upon the completion of the analysis presented above, it is time to synthesize a new sound. The block diagram shown in Figure 8.23 presents the spectral synthesis: transformation and synthesis of a sound which is done in the time domain and frequency domain; generating sinusoids using additive synthesis after transformations; synthesizing the enveloped residual data to a spectral frame after inverse fast Fourier transform(IFFT) with window function; then, applying overlap-add process for each frame.

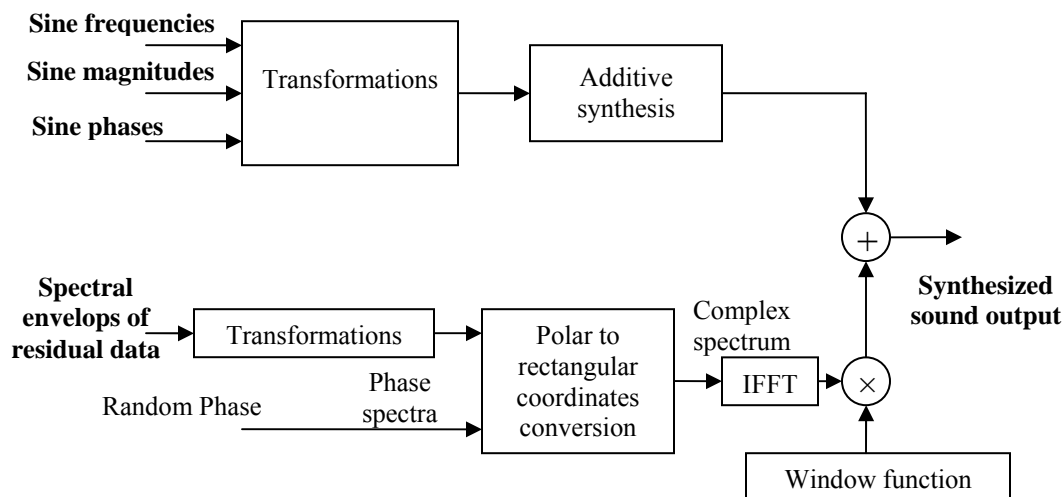


Figure 8.23. Block diagram of the spectral synthesis

The sinusoidal component is generated with additive synthesis in a manner similar to the sinusoidal synthesis that is part of the modeling mentioned above, the difference being that now the phase trajectories might be discarded. Additive synthesis is based on the control of the instantaneous frequency and amplitude of a bank of oscillators. The instantaneous frequencies and amplitudes are smoothly interpolated from frame to frame using linear interpolation to avoid clicks at the frame boundaries.

The synthesis of a stochastic signal from the residual approximation can be understood as the generation of a noise signal that has a frequency and amplitude characteristic described by the spectral magnitude envelopes. It can be applied in the frequency domain by creating a magnitude spectrum from the approximated one, or its transformation, and generating a random phase spectrum with new values at each frame in order to avoid periodicity. An inverse-FFT of the spectral envelopes is applied to generate the stochastic signal.

Once the two components are generated, they must be added together in time domain. Finally, successive frames are combined to obtain the time-varying characteristics of the sound via the overlap-add process.

Since stochastic signal can be considered as a noise signal, the residual parts resulted from spectral modeling synthesis of different kinds of bone drilling sounds are almost same. Therefore, sinusoidal component is the major concern and usually used for sound synthesis of different bone drilling sounds. In this research, drill free-running sound is modeled first to get sinusoidal part and residual part. Then bone drilling sound's characteristics (such as frequency shifts, magnitude changes, etc) are applied to the sinusoidal part. Several major peaks from sinusoidal part are selected for magnitude adjustment using a look-up table, and then frequency shifts are conducted if they exist. Finally following the spectral synthesis procedures shown in Figure 8.23, the new synthesized sound is generated.

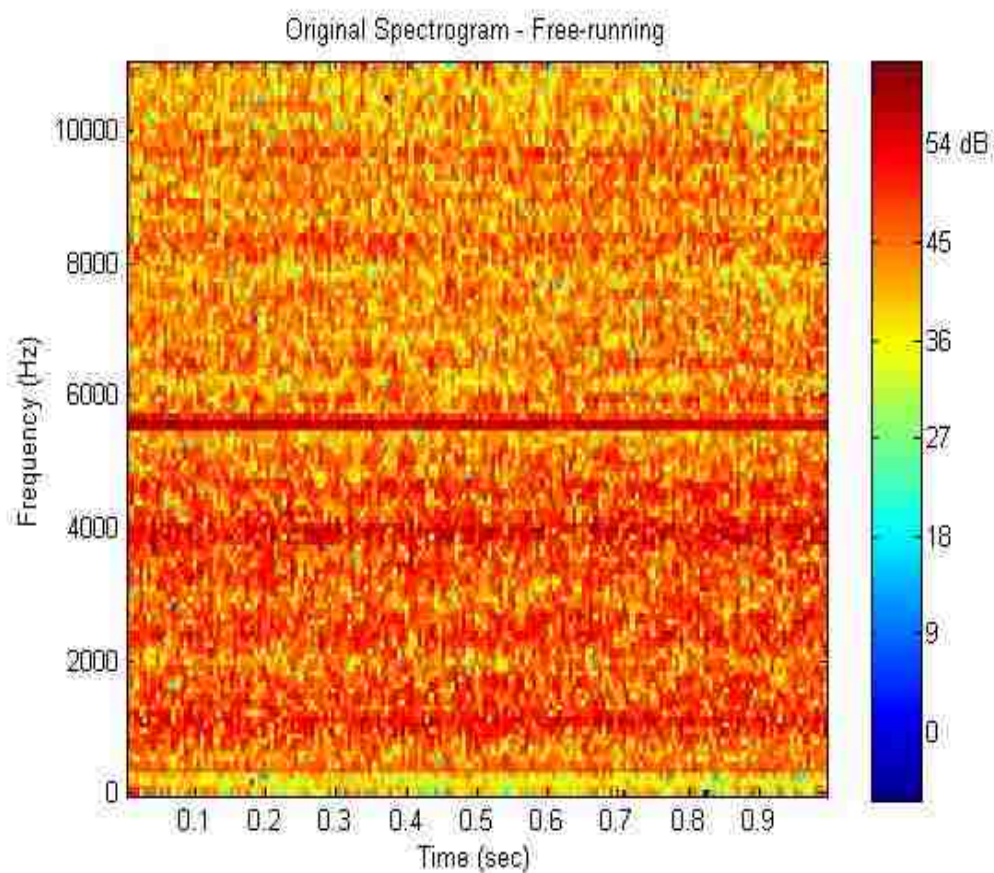
Due to the human ear's listening capability, changes in sound intensity affect human ear's response. Therefore, the logarithmic dB scale change is used as a main threshold for magnitude changes during drilling sound synthesis. Since 1 dB change in sound intensity is barely noticeable and 3 dB-5 dB change has obvious difference in loudness [Basaraba and Archer, 1995], 3 dB differences between two peaks at same frequency is selected as the threshold for the magnitude changes.

Since the virtual drill can be moved around in the virtual environment, several 3D sound effects (such as rolloff, arrival offset, and Doppler shift effects) related parameters are applied to the synthesized sound to obtain more realistic drilling sound. These parameters include the virtual tools' relative position and velocity, and the drilling sound's magnitude and frequency.

8.6.4. Sound Modeling and Synthesis Results. For spectral modeling synthesis, after the calculations of peak detection, peak interpolation, parameter estimation, and peak continuation, the output of the sinusoidal analysis is a set of spectral peak values (frequencies, magnitudes and phases) organized into frequency trajectories, where each trajectory/tracks models a time-varying sinusoid.

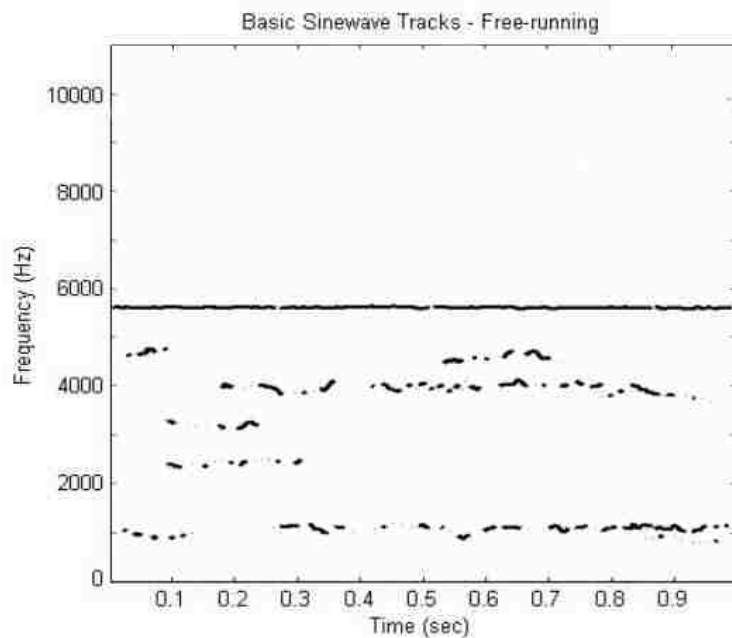
Figure 8.24 shows the results of spectral modeling for free-running sound. Figure 8.24 (a) shows an original spectrogram of a free-running drilling sound. Figure 8.24 (b) shows the basic sine wave tracks after sinusoidal modeling. These tracks are at the frequencies of 5500 Hz, 4600 Hz, 4000 Hz, 3400 Hz, 2400 Hz, and 1100 Hz, which are same to the major PSD peaks frequencies shown in Figure 8.10. Tracks on 5500 Hz seem

to be in a straight line because this frequency is the dominant frequency. The other tracks are not shown as straight lines because they are not dominant and drilling sound is not stationary all the time. Figure 8.24 (c) shows the sinusoidal tracks overlaid on the original spectrogram. Figure 8.24 (d) shows the residual part of the free-running sound shown in spectrogram.

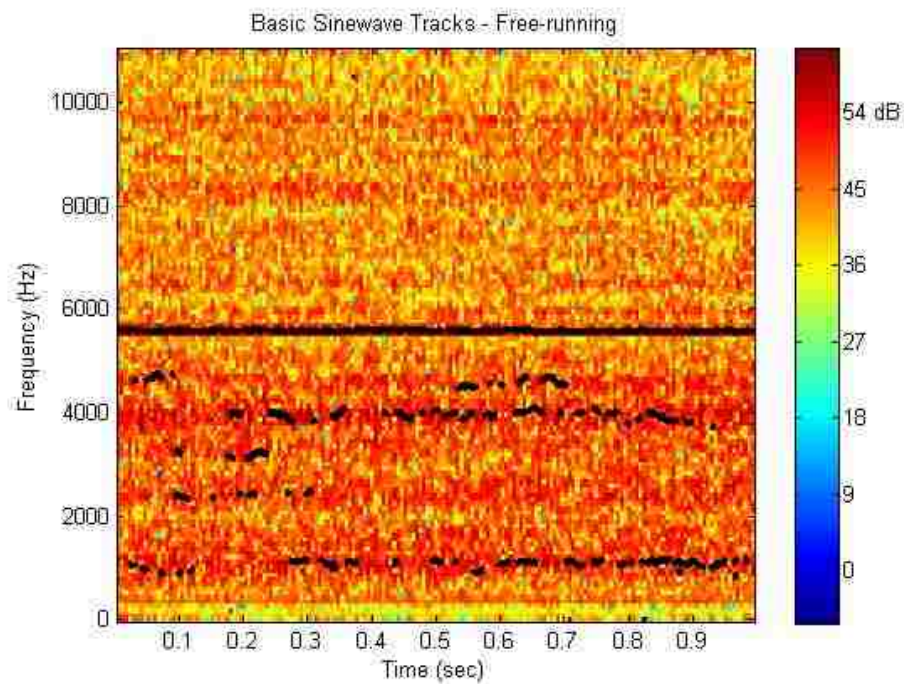


(a)

Figure 8.24. Spectral modeling synthesis results shown in spectrograms: (a) original free-running sound; (b) basic sine wave tracks; (c) basic sine wave tracks overlay; (d) residual part



(b)



(c)

Figure 8.24. Spectral modeling synthesis results shown in spectrograms: (a) original free-running sound; (b) basic sine wave tracks; (c) basic sine wave tracks overlay; (d) residual part (cont.)

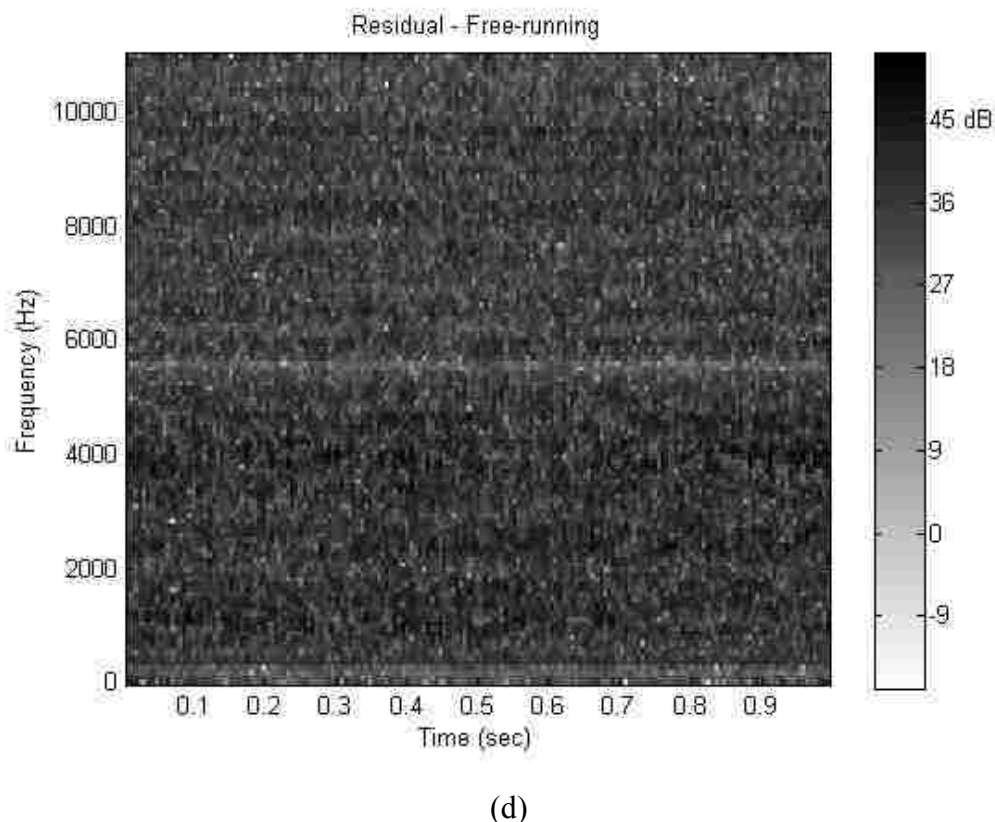


Figure 8.24. Spectral modeling synthesis results shown in spectrograms: (a) original free-running sound; (b) basic sine wave tracks; (c) basic sine wave tracks overlay; (d) residual part (cont.)

Figure 8.25 shows an example of the result of residual approximation for free-running sound. Compared to the sinusoidal analysis and synthesis, the processing of the stochastic part is significantly simpler. Basically the only parameters of the stochastic analysis that can be adjusted are the window length and frame rate.

Figure 8.26 shows the comparisons between original free-running sound and the spectral synthesis sound in the power spectrum (a) and spectrogram (b). The results are shown very good similarities between the original sound and the synthesized sound.

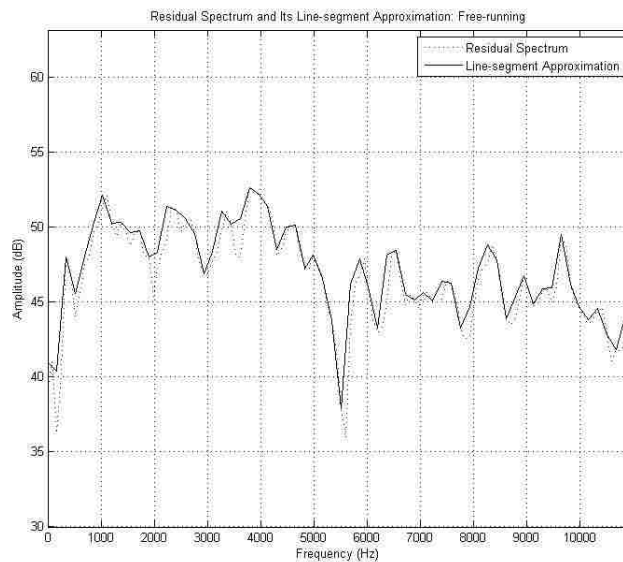
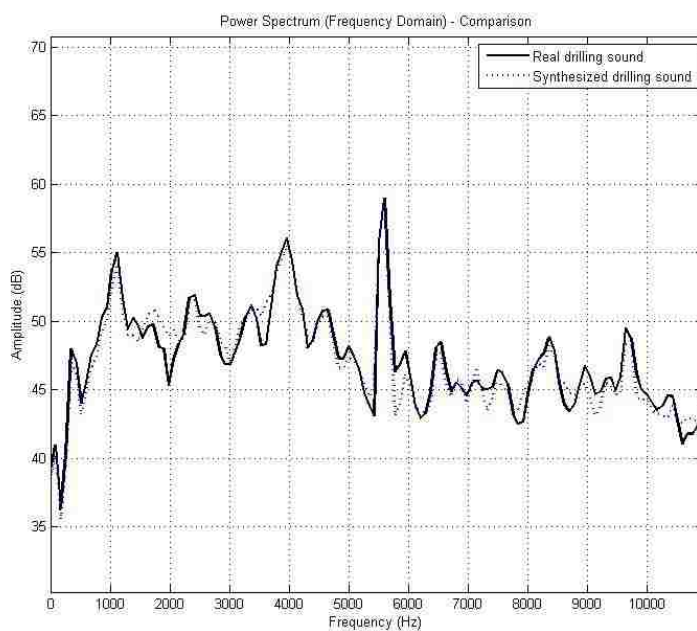


Figure 8.25. Residual spectrum and its line-segment approximation of free-running drilling



(a)

Figure 8.26. Comparisons between original and spectral synthesis sound for drill free-running sound: (a) power spectrum; (b) spectrogram

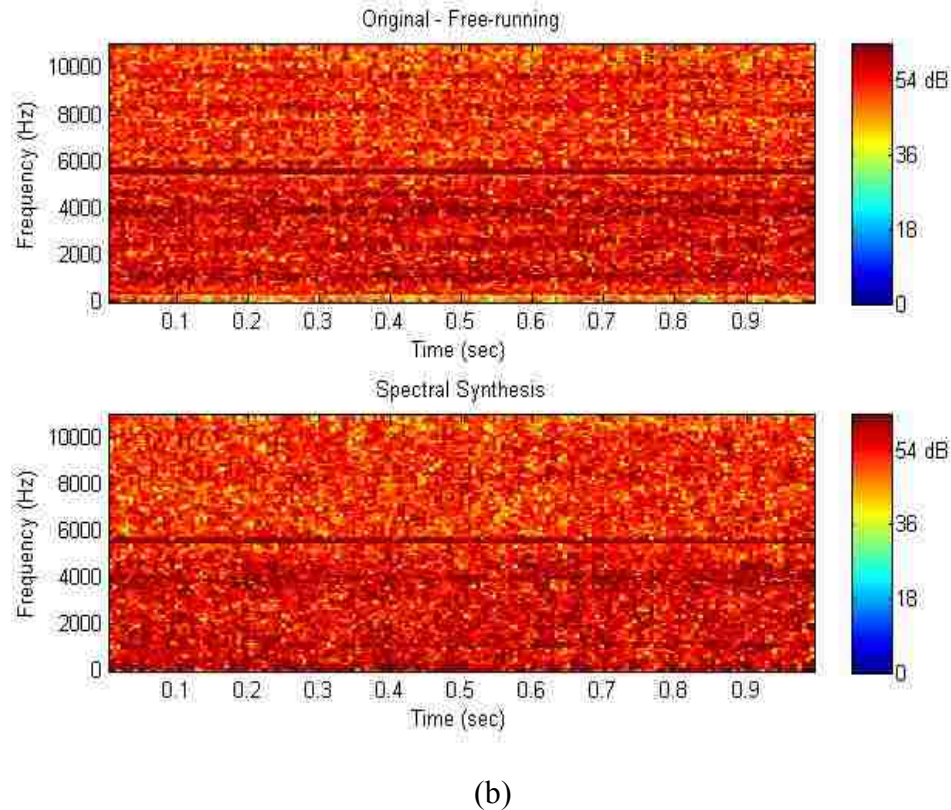
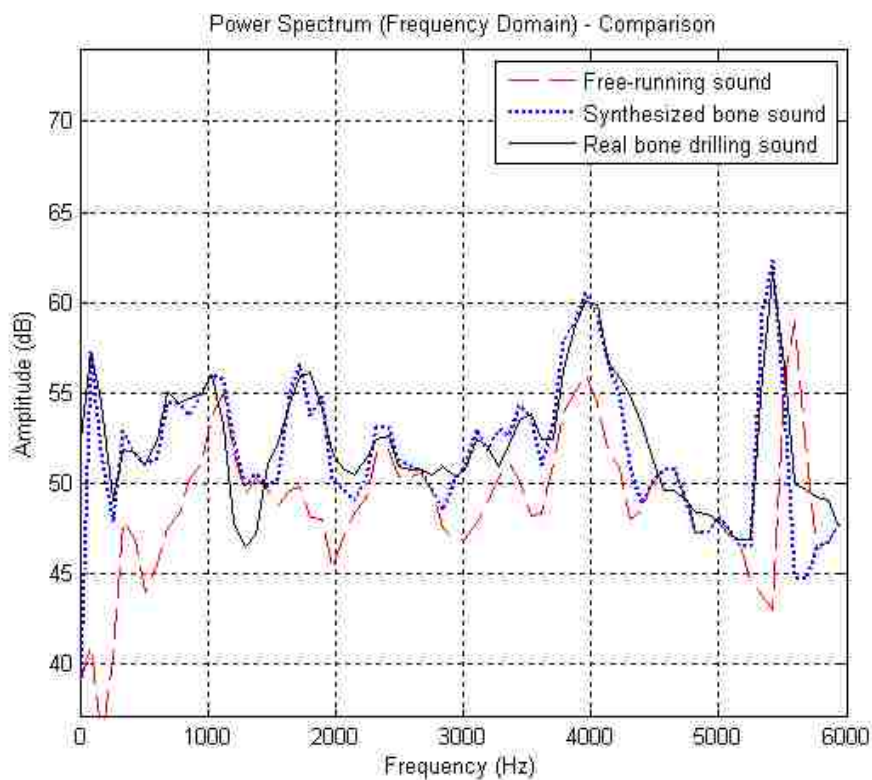
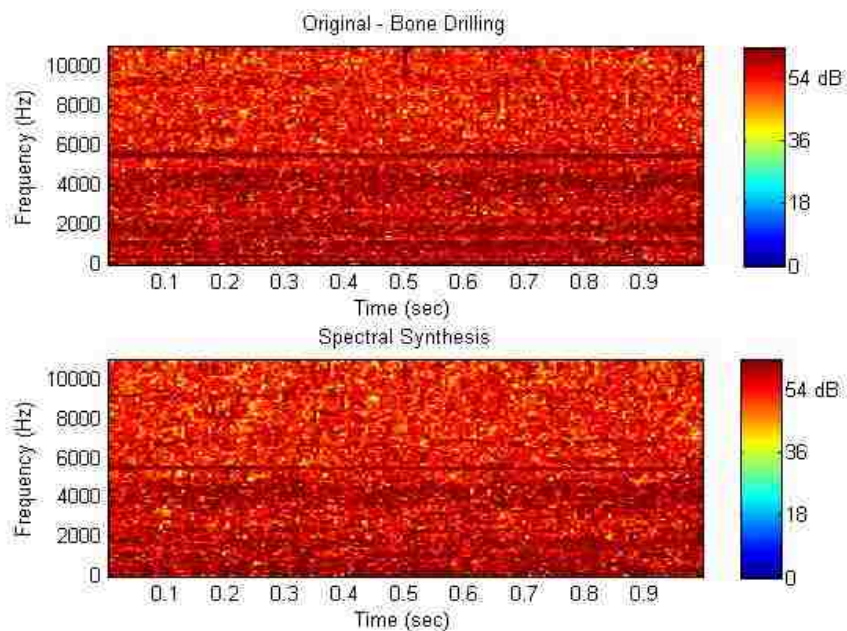


Figure 8.26. Comparisons between original and spectral synthesis sound for drill free-running sound: (a) power spectrum; (b) spectrogram (cont.)

The cortical bone material drilling sound and cancellous material drilling sound can be synthesized based on the drill free-running sound. According to the sound characteristics obtained in Section 8.5, frequency shifts and magnitude adjustments are conducted in sinusoidal part of the free-running SMS model. Since the frequency range of 0-6000 Hz has the major peaks, the sinusoidal synthesis is conducted at that range. The residual part is only considered as a noise signal and it is almost same for the different kinds of bone drilling sounds, therefore only the sinusoidal part is considered as the major concern for current research. Figure 8.27 and Figure 8.28 show the comparisons between real bone drilling sound and synthesized sound for cortical bone drilling and cancellous bone drilling both in power spectrum and spectrogram representations. The results show that the synthesized sounds have very good similarities with the real bone drilling sounds.

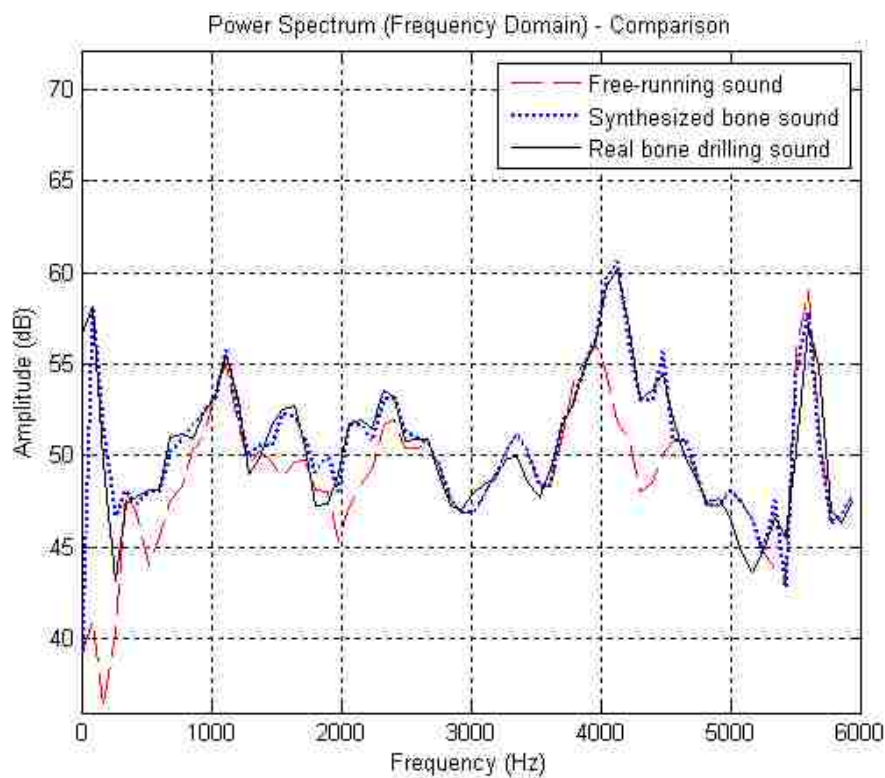


(a)

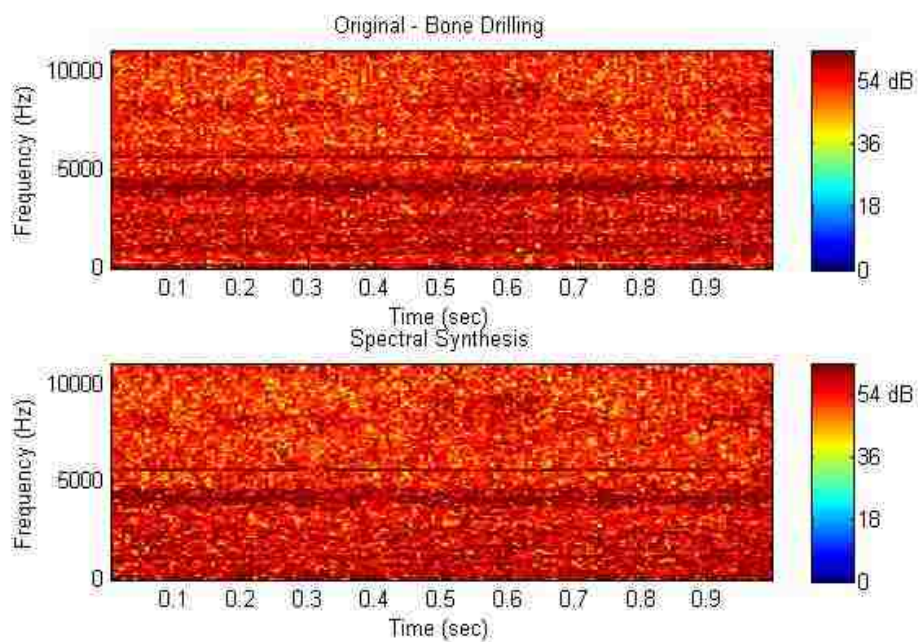


(b)

Figure 8.27. Comparisons between real bone drilling sound and synthesized sound for cortical bone drilling: (a) power spectrum and (b) spectrogram



(a)



(b)

Figure 8.28. Comparisons between real bone drilling sound and synthesized sound for cancellous bone drilling: (a) power spectrum and (b) spectrogram

8.7. SOUND RENDERING

Sound rendering, first introduced by Takala and Hahn [1992], is a technique of generating a synchronized soundtrack for animations in a virtual environment. The synthesized sound in time domain will be used for sound rendering in virtual bone surgery system. Sound rendering outputs the generated sound to the suitable hardware (sound card, loud speaker, etc.), and then the user can hear the sound.

Although sound can be produced through suitable hardware such as sound cards, applying the sound model to the virtual bone surgery system is still an important task after sound model creation. As a Win-32 application, the virtual bone surgery system communicates with sound card and creates sound buffers which exist in hardware such as on-board RAM, wave-table memory, direct memory access channels, or virtual buffers depending on the type of sound card. MS-DirectSound API is chosen as the sound manipulation API for the bone surgery simulation system. The relationships among the bone surgery simulation system, DirectSound API and sound card hardware are depicted in Figure 8.29.

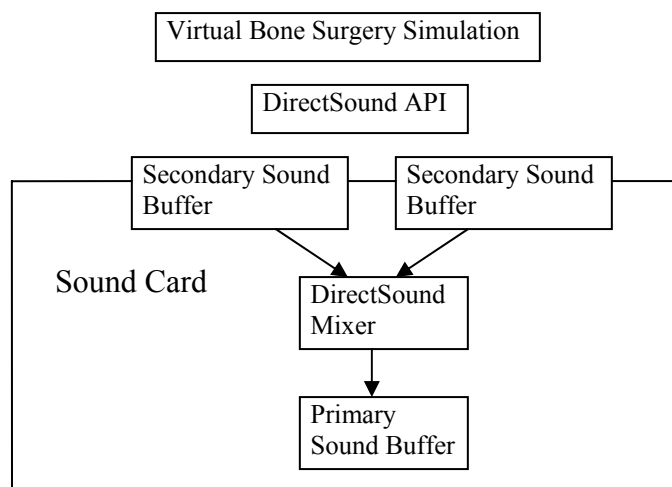


Figure 8.29. DirectSound sound buffering process with bone surgery simulation

The virtual bone surgery system places a set of sounds in secondary buffers, created by the application. DirectSound combines (mixes) these sounds and writes them into a primary buffer, which contains the sound that the listener hears. DirectSound automatically creates a primary buffer, which typically resides in the sound card memory. The virtual bone surgery system creates the secondary buffers in system memory if there is no hardware memory available.

The block diagram of sound rendering calculation procedures is shown in the Section 9. According to the existing multi-point haptic rendering approach, during tool-bone interaction simulation, when a collision is detected, the data of the shapes and attributes (especially the material distribution around the cutter) of the virtual objects are transferred to the sound model for calculation. Sound data in the buffer storage is being continuously updated, and the calculation results are then sent to the audio device driver via our developed sound generation functions to produce the sound.

9. SYSTEM IMPLEMENTATION, INTEGRATION AND RESULTS

9.1. IMPLEMENTATION OVERVIEW

A virtual bone surgery system has been designed, implemented, and integrated using the newly developed methods and techniques on a personal computer. The methods and techniques include medical image processing, large medical data management, geometric modeling, graphic rendering, collision detection, force modeling, haptic rendering, sound modeling, and sound rendering. The overall system is designed with three different modules: Medical Image Processing, Simple Training, and Complex Training, which are integrated in a Windows GUI environment. A well arranged software graphic user interface including menus, icons, and toolbars, as well as a good human-system interaction has been appropriately configured. The challenging task here is to supply users a flexible, extensible, and user-friendly bone surgery system.

The Medical Image Processing module is an important preparation for the bone surgery simulation. The user can load patient-specific CT scan raw data to the system, display them, and analyze them slice by slice. The user can also utilize the system's built-in image processing methods, such as filtering, edge detection, and segmentation to process these CT scans for later surgery purposes. All these processing procedures can be done manually or automatically. The processed data can be saved for later surgery training simulation.

The Simple Training module is used to train surgeons in the basic machining processes (drilling, milling, burring, etc.) and important steps in a typical bone surgery. A learning mode, a practice mode, and a test mode are supplied in this module. In the learning mode, the user can become familiar with the machining processes through videos, animation, pictures, texts, and even through the simulation system itself via demonstrations of how an expert previously performed surgery procedures using this system. In the practice mode, the user can practice different processes by themselves. In the test mode, tests are given for the user to complete in a certain time and results are shown later. The user can load processed CT data, select different tools (machining and non-machining), and change tool tools' geometry, size, and shape from the GUI menu or floating tool bar. The user can choose from two display modes, full screen and Windows-

like display. Practices and tests can be recorded for further study and correction at a later time.

The Complex Training module is similar to the simple training module. It supplies the same functionalities except that it is used for entire surgery procedures, such as knee surgery or hip surgery.

Due to the complexity of the system, a number of C++ classes have been designed. Major ones are shown in UML (Unified Modeling Language) format in Figure 9.1.

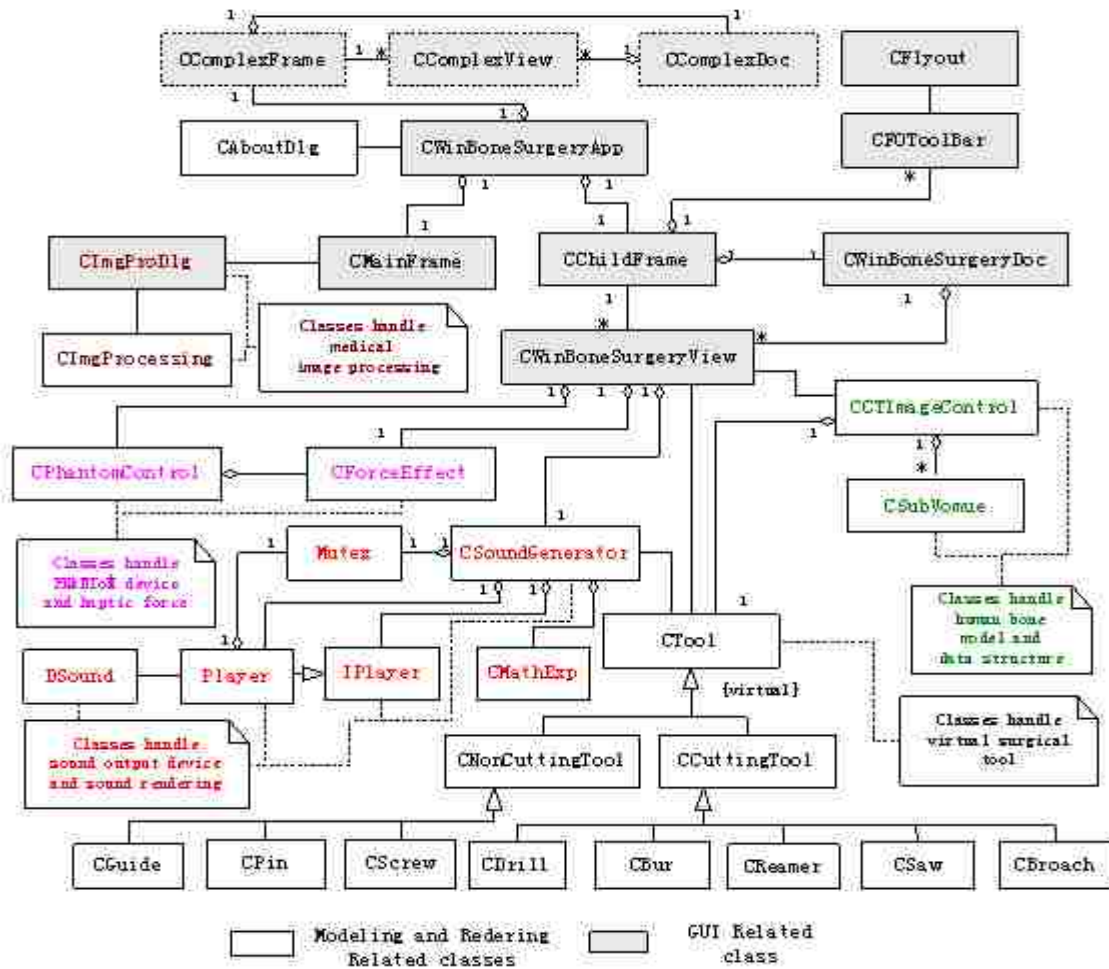


Figure 9.1. Major C++ class diagram in UML format

The relationships of all the major C++ classes for the bone surgery simulation system are illustrated in Figure 9.1:

The C++ classes `CWinBoneSurgeryApp`, `CWinBoneSurgeryView`, `CWinBoneSurgeryDoc`, `CMainFrame`, `CChildFrame`, and `CImgProDlg` are used to construct the whole graphic user interface for MS-Windows environment. To make the simulation system user friendly, the MFC (Microsoft Foundation Classes) library view classes, including Windows common dialogs, ActiveX controls, Windows message processing, and the Document-View architecture are used for implementation and integration. The newly developed C++ classes `CFlyout` and `CFOToolBar` are used to control the Flyout toolbar.

`CImgProcessing` is used for image processing which was mentioned in Section 3. Raw CT data can be pre-processed and the segmentation data can be saved as files for later surgery simulation. A dialog box is implemented as a main GUI for image processing using `CImgProDlg` class.

Classes `CCTImageControl` and `CSubVolume` are used to handle and control the sub-volumes for the human bone model and data structure. `CCTImageControl` is used to read the information from the meta file to construct the input image data files to the proposed data structure, as shown in Figure 5.5, and to fulfill the search and update function. `CSub-volume` is a basic C++ class for volumetric modeling and the display pipeline mentioned in Section 4 and for the Quad-tree subdivision and data structure mentioned in Section 5. Each sub-volume shown in Section 5.5 is an object of this class.

`CTool`, `CCuttingTool`, `CNonCuttingTool`, `CDrill`, `CBurr`, and `CGuide`, are used to represent the virtual surgical tool. In addition to the methods mentioned in Section 6.1, inheritance and polymorphism are applied by using C++ for implementation. Class `CTool` is a pure virtual class and virtual functions are used for all the other inherited classes.

`CPhantomControl` is used for the PHANToM device and `CForceEffect` is for the different PHANToM force effects, such as drilling force and burring force mentioned in Section 7.

`CSoundGenerator`, `IPlayer`, and `Player` are used for the sound output device handling and sound rendering mentioned in Section 8.

9.2. MULTI-THREADING WITH MULTI-RATE

To enhance simulations' realism, a haptic interface is necessary to provide the user with a realistic feel of the surgical force, in addition to graphic visualization and sound auditory during the simulation of tool-bone interaction. A major challenge of graphic display, haptic display, and auditory display in a virtual bone surgery is to simultaneously satisfy the stringent requirements of 1,000 Hz update rate for haptic rendering to provide a continuous feel of force, 30 Hz update rate for graphic rendering to get realistic visualization of the surgical operation, and 22k Hz sound sample rate for sound rendering to get good auditory. Due to frequent model updates and the large amount of calculation required for display, haptic and sound rendering, parallel processing should be used to speed-up the computation.

Multi-threading is applied in this system because the threading technique has two main advantages over the message passing technique. First, multi-threading has a drastically smaller parallel programming overhead (no explicit exchange of messages, no buffer [size] checking, no buffer overflow). Second, it will provide the surgery simulation system with the possibility of a high-end multi-processor environment. All implemented threads can run "simultaneously" during bone surgery simulation to avoid the following sources of delay:

1. Computational delay: time elapsed while the data is in the host system and while the system does computations,
2. Rendering delay: time elapsed while the graphics engine generates the result picture,
3. Display delay: time elapsed between sending images to the display and the display actually showing them,
4. Synchronization delay: time in which data is waiting between stages without being processed,
5. Frame-rate-included delay: between two frames the display is not updated, causing the user to see an outdated image stream. This display can also be considered a special case of synchronization delay between the display system and the human eye.

Figure 9.2 shows a block diagram of multi-threading in the virtual bone surgery simulation. These threads include:

- **User interface thread** - a user interface thread that performs the most user interface related work in a message map to respond to events and messages generated by user interaction with the application.
- **Simulation thread** - a main worker thread that does the most computational work in a 'while' loop. Executes tasks including initialization (bone sub-volume modeling, data structure construction and tool modeling) and loop calculations (reading of the haptic device's stylus position and orientation, collision detection, force calculation, and model update if possible).

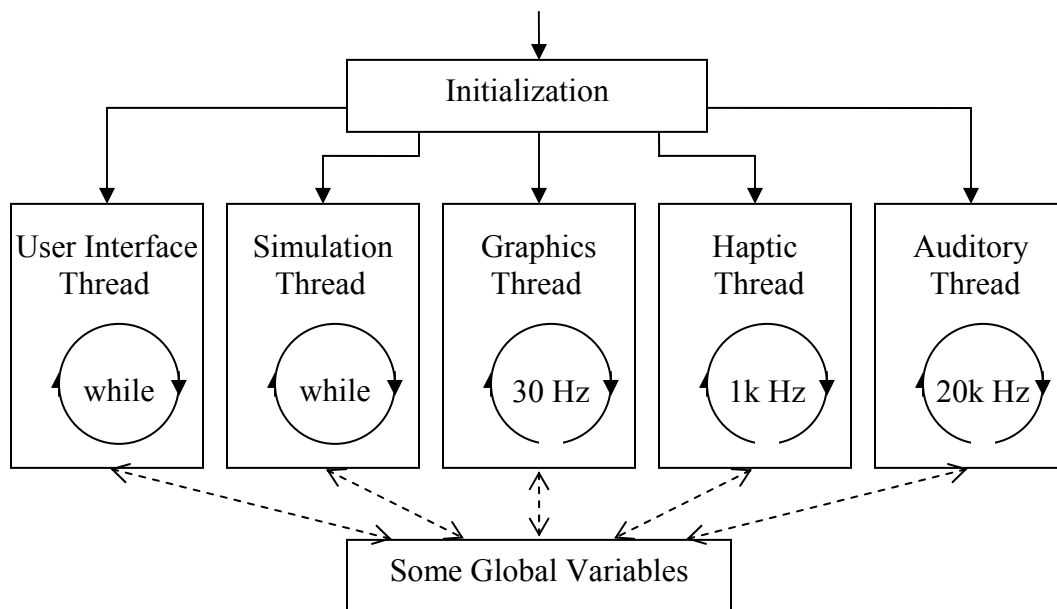


Figure 9.2. Block diagram of multi-threading in the virtual bone surgery simulation

- **Graphic thread** – a worker thread with a 30 Hz timer in it executes graphic display implemented using the advanced marching cube algorithm for surface rendering of the voxels and active sub-volume(s).

- **Haptic thread** - a worker thread with a 1k Hz timer in it performs collision flag checking and sends the calculated interface force signal to the API (Application Program Interface) functions of the haptic device.
- **Auditory thread** - a worker thread with a 20k Hz timer in it performs collision flag checking and sends the calculated interface sound signal to the API (Application Program Interface) functions of the sound related hardware, e.g., sound card.

Since multiple threads can access same set of static global data, including active sub-volume queue and collision detection flag, the simulation program must provide a way to avoid possible resource conflicts. For example, the simulation thread might update and pushback the active sub-volumes to the active sub-volume queue, but the graphic thread popfronts the sub-volume from the active sub-volume queue, so this problem must be avoided by using semaphores to control access to the structure. A mutex (short for *mutual exclusion*) is a way of communicating among threads or processes that are executing asynchronously. This type of communication is typically used to coordinate the activities of multiple threads or processes by controlling access to a shared resource by “locking” and “unlocking” the resource. To solve this *active sub-volume queue* update problem, the simulation thread could set a mutex indicating that the data structure is in use before performing the update, which would clear the mutex after the active sub-volume queue has been processed. The graphic thread must wait for the mutex to be clear before updating the rendering. This process of waiting for a mutex is often called “blocking” on a mutex because the process is blocked and cannot continue until the mutex clears.

Figure 9.3 is a brief flowchart of multithreading calculation for simulation thread, haptic thread, graphics thread and auditory thread. Detailed explanations of the methods and techniques of collision detection, force calculation, graphic rendering, scalars change, and data model update are given in previous sections.

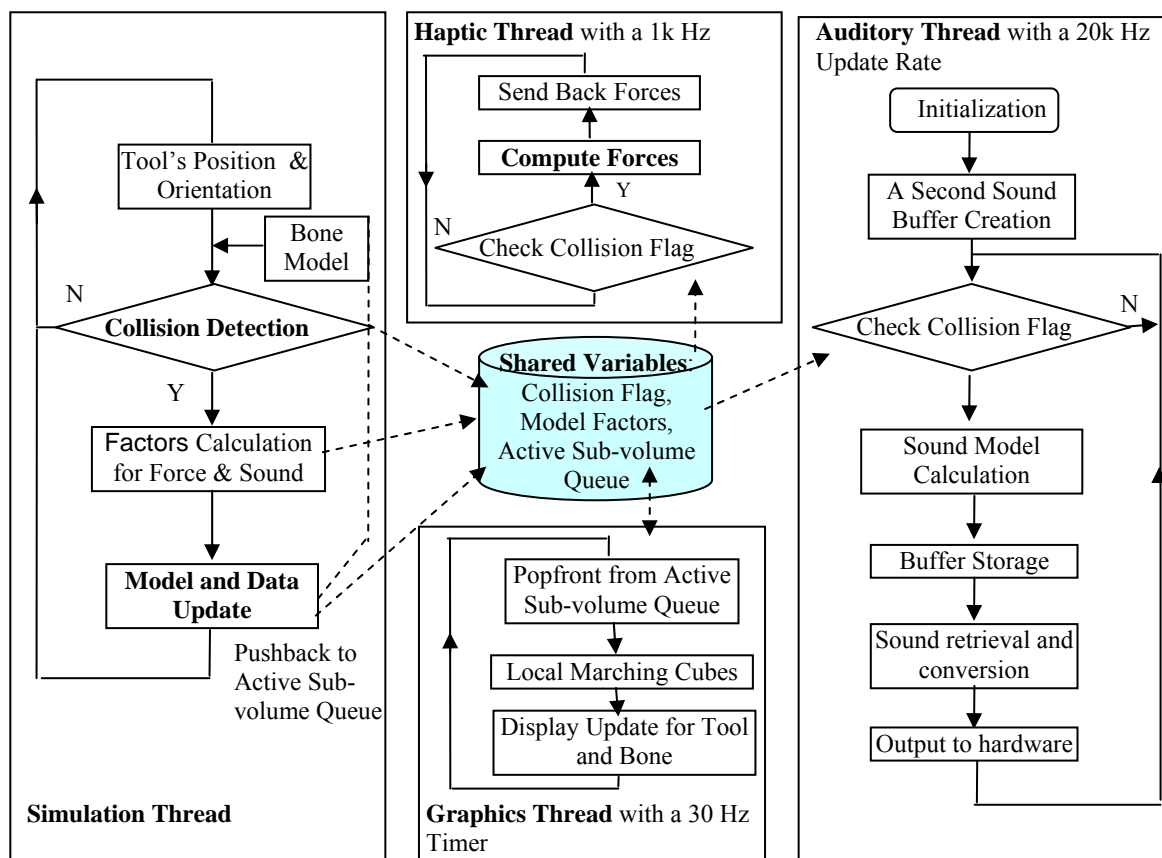


Figure 9.3. Flowchart of multi-threading for simulation thread, haptic thread, graphics thread, and auditory thread

9.3. GUI OF THE VIRTUAL BONE SURGERY SYSTEM

Based on the requirements and functions of the virtual bone surgery system, graphic user interface (GUI) has been designed and developed for different modules. All the methods and technologies mentioned above are implemented and integrated under such a user interface.

9.3.1. Main GUIs for Different Modules. Figure 9.4 shows the graphic user interface of the Medical Image Processing module. It consists of a medical image display area, a slider bar for selecting the current display slice on the left of the layout, and file loading, image-preprocessing, image segmentation, root sub-volume selection on the right. The user can load any CT scan raw data, pre-process the loaded image data with

Gaussian filter, Median filter, or without filter, then use window and level to emphasize the interested area, segment the pre-processed image with manual or automatic methods slice by slice, and output the root-sub-volume to files for later bone surgery simulation.

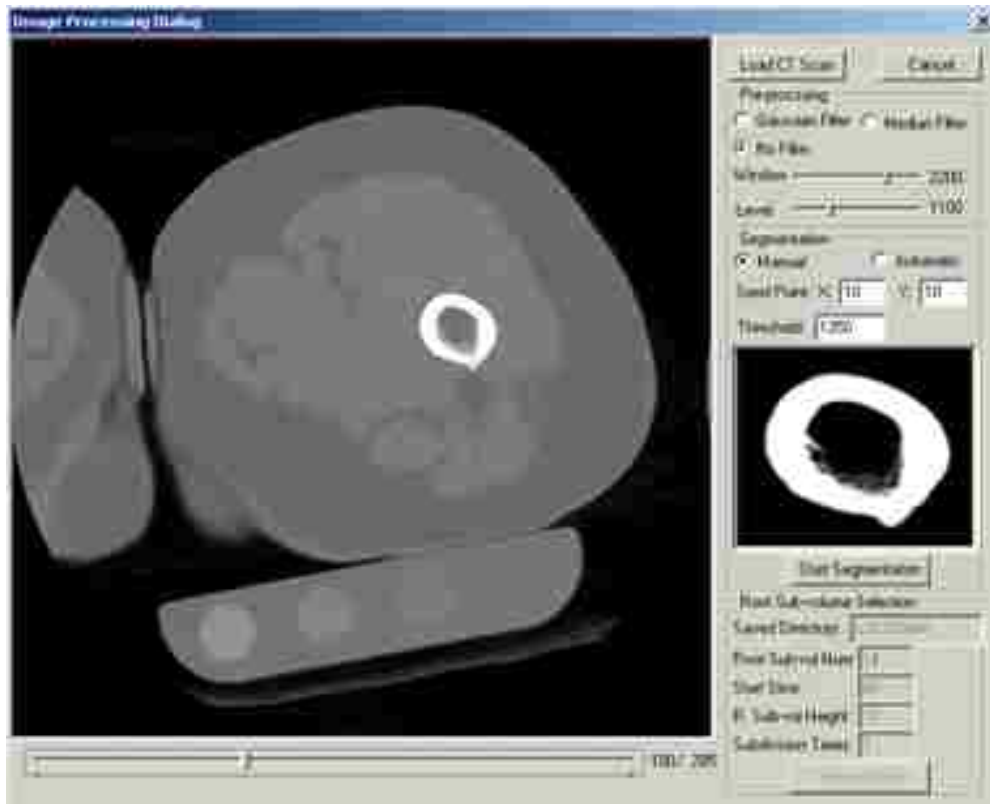


Figure 9.4. Graphic user interface of the image processing module

The layout of the Simple Training Module GUI is shown in Figure 9.5. It includes a menu bar for the Simple Training Module, an upper toolbar, a left toolbar for the groups of the surgical tools (the cutting tools and the non-cutting tools), a lower toolbar for the current tool's information, and a status bar which shows the current PHANToM position. In the main working area, the user can manipulate the PHANToM device to control the virtual surgical tool and perform cutting processes such as drilling, reaming, sawing, as well as non-cutting processes.

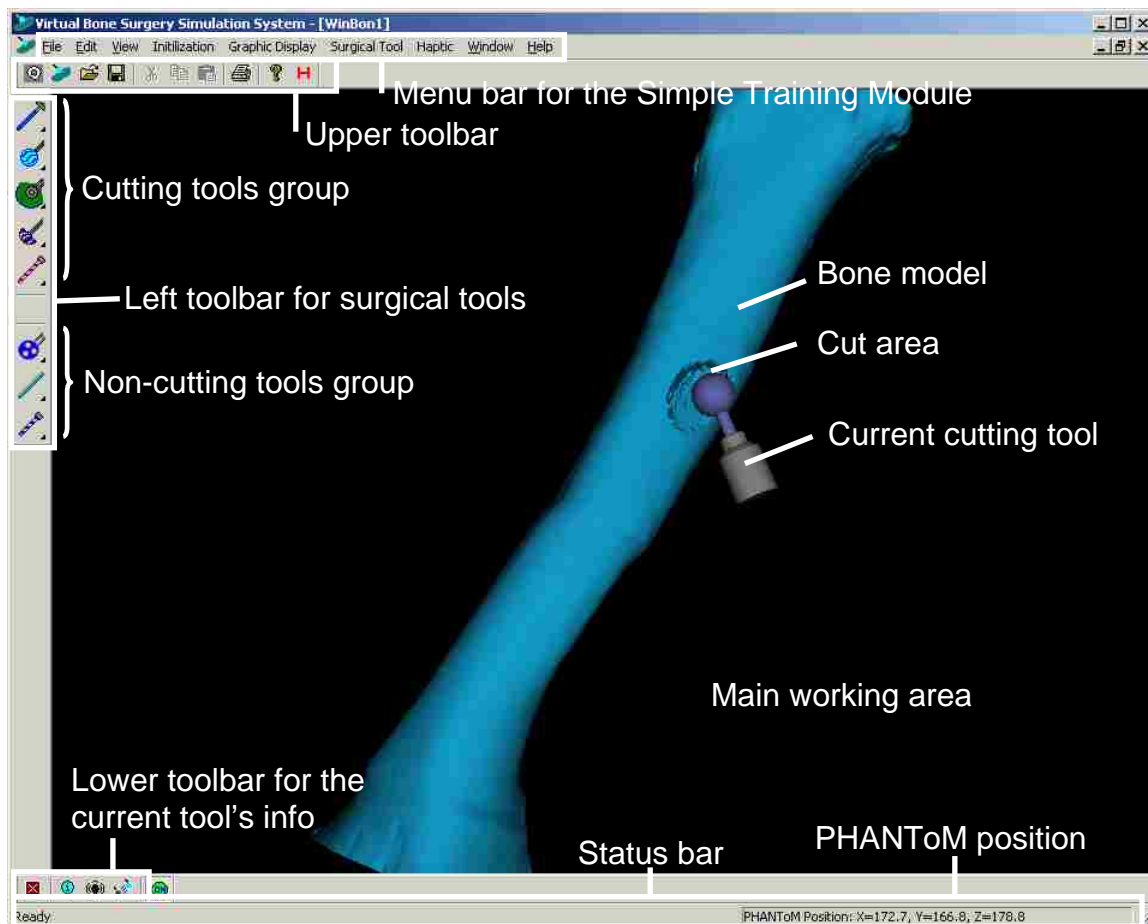


Figure 9.5. Graphic user interface of the simple training module

The left toolbar is a basically a tool box containing various surgical tools for selection in virtual bone surgery. It is a Flyout toolbar which allows the user choose a sub-option of a more “general” action. If the user clicks and holds the displayed button longer, a “Flyout” toolbar showing all the sub-options will pop-up next to the displayed button on the toolbar. If the user chooses a different sub-option from the Flyout toolbar, the button on the main toolbar is automatically updated. Table B.1 in Appendix B shows all the options for the cutting tools and non-cutting tools. The first five groups are cutting tools, and the last three groups are non-cutting tools.

The GUI of the Complex Training Module is almost the same as that of the Simple Training Module. It supplies the same functionalities except that it is used for the entire surgery procedures, such as knee surgery or hip surgery.

9.3.2. Other GUIs for the Parameters Selection. In addition to the main GUIs mentioned above, there are also some other GUIs for the parameters selection. These GUIs are usually designed and implemented using Microsoft MFC model dialog.

Figure 9.6 shows the GUI for changing the properties of the current surgical tool. These properties include tool diameter, spindle speed, tooth number, and feed rate. With changes to different parameters, graphic rendering, haptic rendering, and even cutting sound will change accordingly.

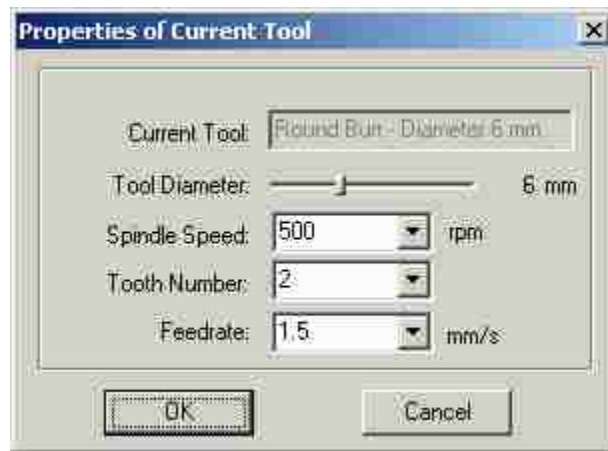


Figure 9.6. GUI dialog for the current tool's properties change

Figure 9.7 shows the rendering sound control and real-time wave graph display. During simulation the user can control the volume of the simulated sound and select whether or not the sound is played. The computation time for sound rendering is displayed in milliseconds at the top of the wave graph.

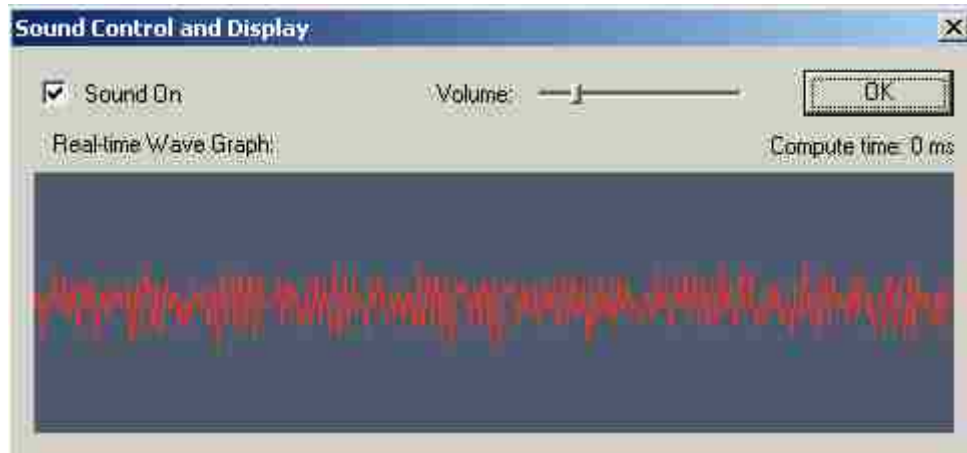
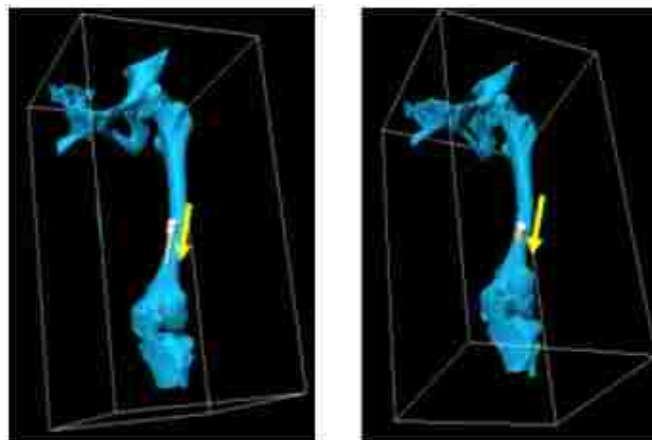


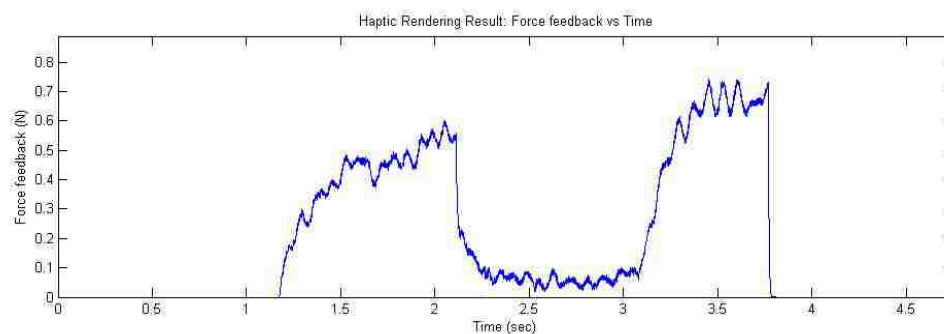
Figure 9.7. GUI dialog for sound control and display

9.4. SIMULATED MATERIAL REMOVAL PROCESSES

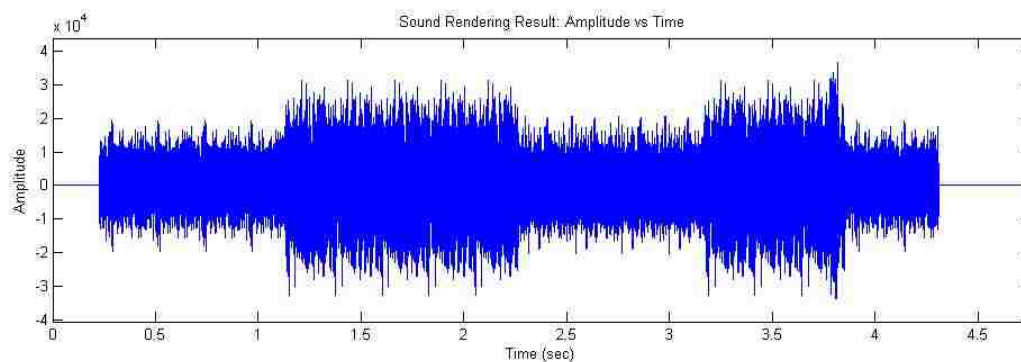
Using the newly-developed virtual bone surgery system, the user can navigate in the virtual 3-D scene and do drilling operations on the bone model. The bone model simulates a real bone structure: its boundary simulates the cortical bone structure, and its interior simulates the trabecular bone structure. In our bone surgery simulation system, the user can manipulate the virtual tool to carry out the desired hole drilling on the virtual bone. For example, the user can operate the virtual drill from empty-run drilling to cortical bone structure drilling, then trabecular bone drilling, and back to cortical bone drilling again, and finally empty-run drilling. Figure 9.8 (a) shows the graphic rendering of the simulated free drilling procedure. Figure 9.8 (b) shows these changes from the drilling force profile obtained from simulated bone drilling. The drilling force output in the simulation system was scaled to the output range of the haptic device. The result shows that the force profile is very close to the force profile in the real-world drilling operation reported by Allotta et al. [1996]. Figure 9.8 (c) shows the sound rendering result of free drilling. It is very clear that the simulated visual effects, haptic force and the simulated sound can be changed simultaneously according to the user's action.



(a)



(b)



(c)

Figure 9.8. Illustration of free drilling: (a) material removal; (b) simulated drilling force profile; (c) simulated drilling sound profile

Some procedures in orthopedic surgery require fixation of the implant, in which bone-screws are put in proper positions to anchor further operations. Such a task relies on

accurate placement of those screws. Guided drilling can guarantee accurate screw positioning and hole making. In the simulation process, the virtual drill guide is gripped and put in the proper position for the user to perform virtual drilling using the PHANTOM device. After positioning the guide, the drill can move only in the direction indicated by the guide. Figure 9.9 provides two examples of guided drilling simulation: (a) and (c) illustrate gripping and positioning the guide; (b) and (d) illustrate drilling with the guide.

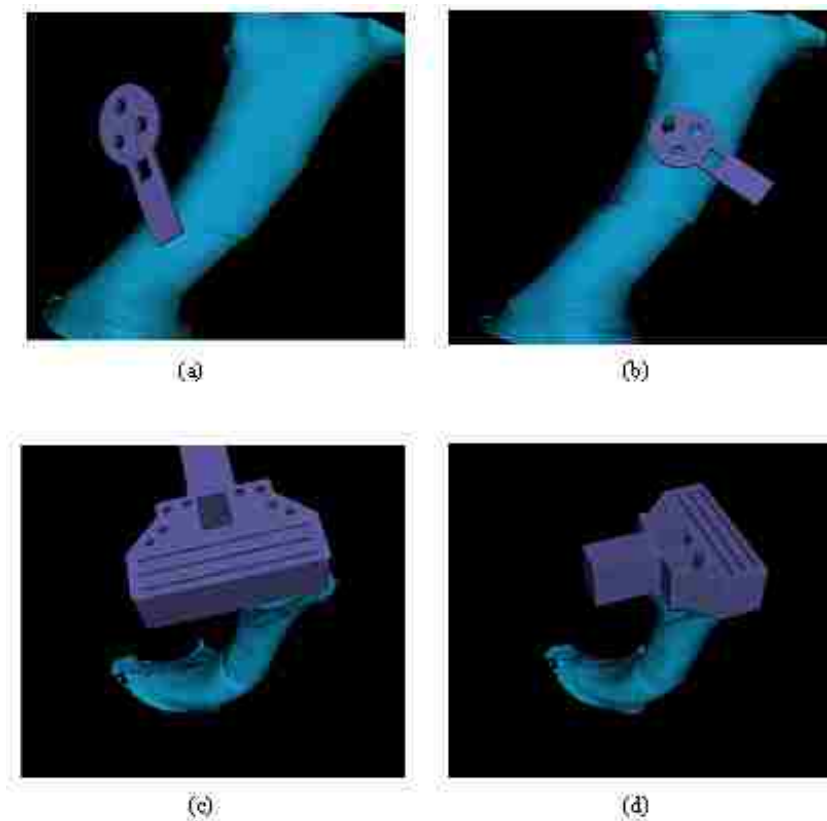


Figure 9.9. Illustration of guided drilling

Figure 9.10 illustrates some other machining processes such as surface burring and milling. The force profiles of these two processes can be found in the Section 7.

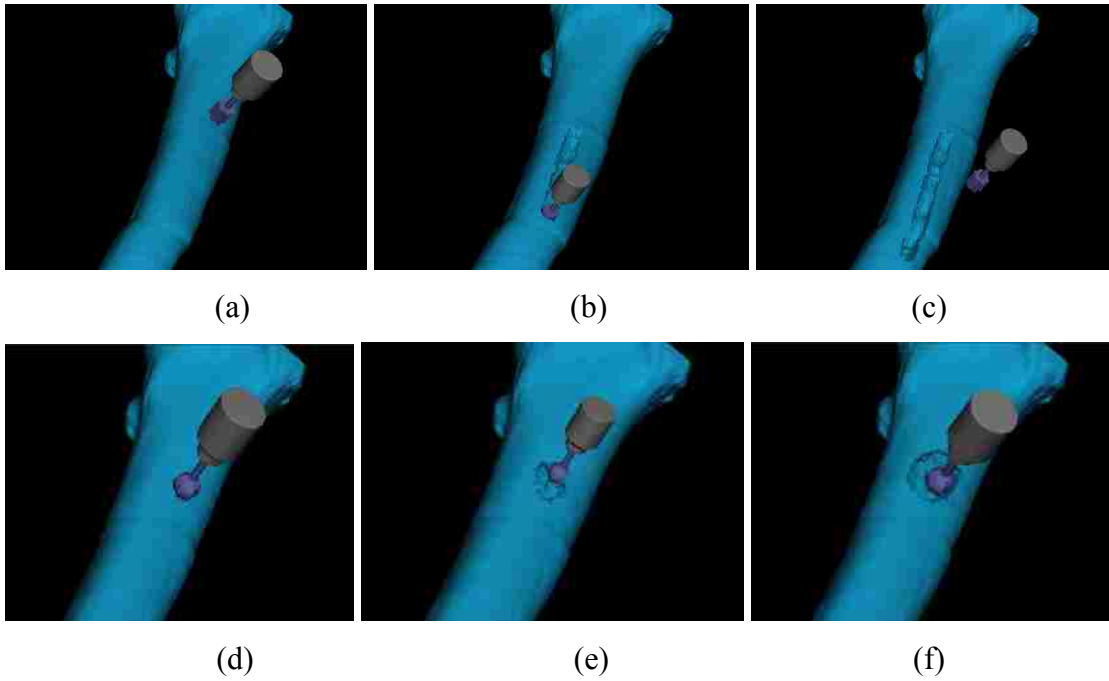


Figure 9.10. Some screen shots of the machining processes: (a)-(c) before/during/after milling; (d)-(f) before/during/after surface burring

10. CONCLUSIONS AND FUTURE WORK

10.1. CONCLUSIONS

The objective of this dissertation was to build a prototype virtual bone surgery system which can be used for training in orthopedic surgery, as well as for planning and rehearsal of bone surgery procedures. Towards the development of such a real-time interactive simulator, the modeling and rendering techniques and methods were presented and developed.

This dissertation describes image processing, geometric modeling, graphic rendering, large data management, modeling and representation of surgical tools, force modeling and haptic rendering, sound analysis, sound modeling, and sound rendering techniques and methods. The results were demonstrated on some real-world human bone models utilizing graphical, haptic, and auditory interfaces.

In the developed system, the user can perform virtual bone surgery by “seeing” bone material removal through a graphic display device, “feeling” the force via a haptic device, and “hearing” the sound of tool-bone interaction simultaneously.

The main contributions of this work include:

1. A large medical data management method. In addition to medical data acquisition and image processing, bounding volume and quadtree adaptive subdivision are applied to remove irrelevant data and to organize the remaining data, considering the implementation complexity, memory storage, and computational overhead. A special efficient data structure is proposed to store and organize the image data, which is processed prior to simulation to reduce the data processing time and update time during simulation. Using these techniques, the computer memory requirement and the data processing time are reduced drastically, enabling real-time bone surgery simulation. Compared to the more commonly used octree subdivision technique, this method is more suitable for surgery involving long bones.

2. A collision detection algorithm and force calculation method for haptic rendering. This dissertation has proposed a volumetric-based multi-point collision detection algorithm to overcome the difficulty of unwanted jerky force feedback computation during bone surgery simulation. Collision detection accuracy is increased to

half-voxel size. Cutting force calculation is also presented as a means of obtaining both haptic force direction and magnitude and at the same time enhancing the realistic feel of the surgical force.

3. Sound analysis, modeling, and rendering methods. In order to introduce sound effects to the virtual bone surgery system, this dissertation has presented the techniques and methods of sound acquisition, analysis, modeling and rendering. Based on the sounds captured during the drilling experiments on different artificial bone materials, analysis is conducted in both time domain and frequency domain in order to find the sound characteristics of bone drilling. Spectral subtraction is used to obtain the drilling sound difference caused by tool-bone interaction. A spectral modeling synthesis method is applied to develop the sinusoidal model and residual model, which are then used for auditory rendering in the bone surgery simulation.

4. Multi-threading with multi-rate. To include both graphic display, haptic display, and auditory display in a virtual bone surgery system and to simultaneously satisfy the stringent requirements of different update rates for graphic, haptic, and auditory rendering, multi-threading with multi-rate computational environment is constructed. To make the simulation system more intuitive and interactive, several threads run with multiple rates in parallel: a simulation thread (main worker thread performing most computations), a graphic thread (worker thread with a 30 Hz timer to fulfill graphic display with surface rendering), a haptic thread (worker thread with a 1kHz timer to perform force rendering), an auditory thread (worker thread with a 20kHz sound sample rate for sound rendering).

10.2. FUTURE IMPROVEMENTS

A virtual bone surgery system has been developed in this work. Although this system has addressed some challenging modeling and rendering issues, there are still many aspects that are worthy of further research. Some possible future research tasks are described below:

1. It may be possible to further develop the virtual bone surgery system by including more surgical procedures. Interaction models between different surgical tools and the bone model can be further investigated too.

2. The PHANToMTM device, used as a haptic device in this work, is only capable of exerting the maximum force of 6.4N and the output force only has 3-degree of freedom. A different haptic device could supply more realistic haptic feedback.

3. In this research, only one kind of drill and steady drilling speed and feed rate are used for different kinds of bone materials during sound acquisition, analysis and auditory rendering. It would be necessary to consider different parameters for drills in later work.

4. It may be possible to add some auxiliary effects to the simulation system, such as bleeding, debris formation, and fluid flow to enhance the realism of the simulation.

5. Evaluation of the benefits of the multimodal simulator by surgeons and human computer interaction experts would help verify the practicality of the developed surgery simulator.

APPENDIX A.
MAIN PROCEDURES OF KNEE SURGERY AND HIP SURGERY

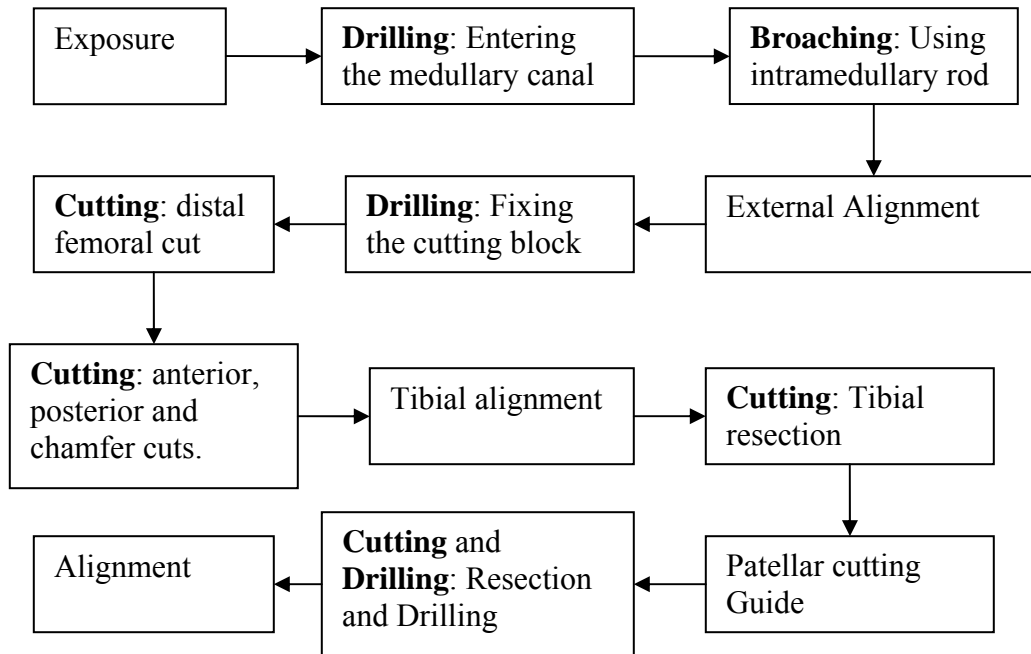


Figure A.1. Main procedures of knee surgery (primary cruciate-retaining)

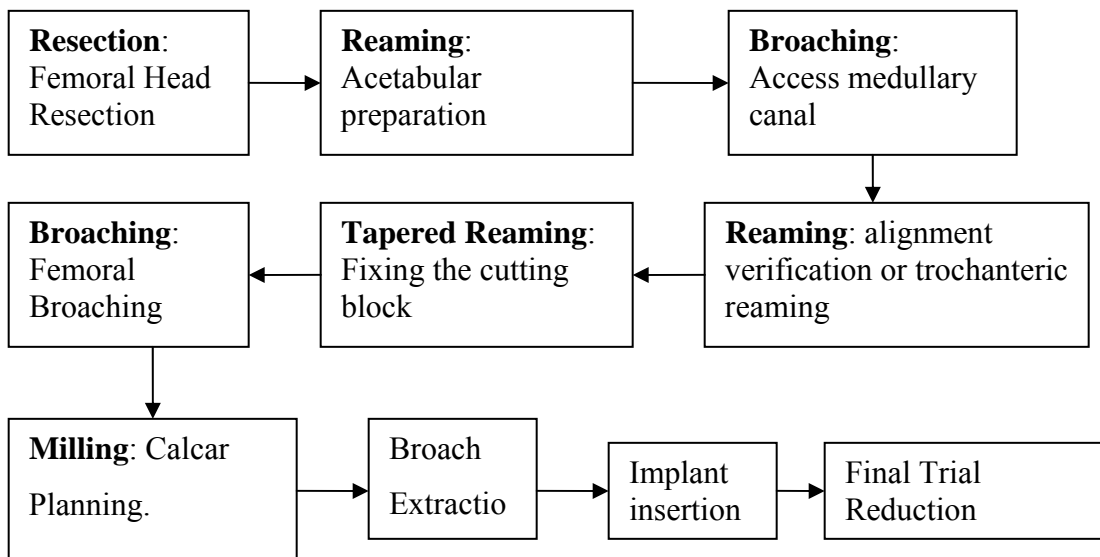


















Figure A.2. Main procedures of hip surgery (total hip replacement)

APPENDIX B.
FLYOUT TOOLBAR FOR VIRTUAL SURGICAL TOOLS

Table B.1. Flyout tool bar for surgical tools

Current Tool Displayed	Flyout Tools Displayed	Descriptions
		Drill (1/8" Drill, 5/16" Drill, End Drill)
		Bur (Round Bur, Barrel Bur, Contoured Cutting Bur)
		Saw (Oscillating Saw1, Oscillating Saw2, Oscillating Saw3, Reciprocating Saw)
		Reamer (Sphere Reamer, Cylindrical Reamer, Cylindrical Outside Reamer)
		Broach (Broach 1, Broach 2)
		Guide (Guide 1, Guide 2, Guide 3)
		Pin (Pin 1, Pin 2, Pin 3)
		Screw (Screw 1, Screw2, Screw 3)

BIBLIOGRAPHY

- Abouzgia MB, James DF (1995) Measurements of shaft speed while drilling through bone. *Journal of Oral Maxillofacial Surgery*, 53: 1308-1315.
- Adrien JM (1991) The missing link: modal synthesis. In : Poli GD et al. (eds), *Representations of Musical Signals*, The MIT Press, Cambridge, Massachusetts, USA, pp 269-297.
- Agus M, Giachetti A, Gobbetti E, Zanetti G, Zorcolo A (2002) Real-time haptic and visual simulation of bone dissection. *IEEE Virtual Reality Conference*, pp 209-216.
- Allotta B, Belmonte F, Bosio L, Dario P (1996) Study on a mechatronic tool for drilling in the osteosynthesis of long bone: tool/bone interaction, modeling and experiments. *Mechatronics* 6(4): 447-459.
- Astley O, Hayward V (2000) Design constraints for haptic surgery simulation. *Proceedings of the 2000 IEEE international conference on Robotics & Automation*, San Francisco, CA.
- Avila RS, Sobierajski LM (1996) A haptic interaction method for volume visualization. *IEEE Visualization Proceedings*, San Francisco, MA, pp 197-204.
- Bærentzen A (1998) Octree-based volume sculpting. *Proceedings of IEEE Visualization Conference*, Research Triangle Park, NC, pp 9-12.
- Bærentzen A (2001) Volume sculpting: intuitive, interactive 3D shape modeling. *IMM*, May 15.
- Barker VL (1999) Cathsim. In: Westwood JD, Hoffman HM, Robb RA, Stredney D (eds), *Proceedings of Medicine Meets Virtual Reality*, IOS Press, San Francisco, USA, pp 36-37.
- Basaraba MB, Archer JA (1995) *IPT's rotating equipment handbook: machinery reliability & condition monitoring*. IPT Publishing & Training Co.
- Benko U, Petrovcic J, Juricic D, Tavcar J, Rejec J, Stefanovska A (2004) Fault diagnosis of a vacuum cleaner motor by means of sound analysis. *Journal of Sound and Vibration*, 276: 781-806.
- Bentzen SM, Hvid I, Jorgensen J (1987) Mechanical strength of tibial trabecular bone evaluated by X-Ray Computer Tomography. *Journal of Biomechanics* 20(8): 743-752.
- Berkley J, Weghorst S, Gladstone H, Raugi G, Berg D, Ganter M (1999) Fast finite element modeling for surgical simulation. In: Westwood JD, Hoffman HM, Robb RA, Stredney D (eds), *Proceedings of Medicine Meets Virtual Reality* IOS Press, San Francisco, USA, pp 55-61.

- Berouti M, Schwartz R, Makhoul J (1979) Enhancement of speech corrupted by acoustic noise. Proceedings of the IEEE International Conference on Acoustic, Speech, and Signal Processing, pp 208-211.
- Bloomenthal J, Bajaj C, Blinn J, Cani-Gascuel M, Rockwood A, Wyvill B, Wyvill G (1997) An introduction to implicit surfaces. Los Altos, CA: Morgan Kaufmann Publishers.
- Boada I, Navazo I (2001) Multiresolution isosurface fitting on a surface octree. 6th International Fall Workshop Vision, Modeling and Visualization 2001, Stuttgart, Germany, pp 318-324.
- Boll S (1979) Suppression of acoustic noise in speech using spectral subtraction. IEEE Transactions on Acoustics, Speech, and Signal Processing, ASSP-27(2): 113-120.
- Borro D, Garcia-Alonso A, Matey L (2004) Approximation of optimal voxel size for collision detection in maintainability simulations within massive virtual environments. Journal of the European Association for Computer Graphics, Computer Graphics Forum, 23(1): 13-23.
- Brice R (1997) Multimedia & virtual reality engineering. Newnes, Oxford.
- Bro-Nielsen M, Helfrick D, Glass B, Zeng X, Connacher H (1998) VR simulation of abdominal trauma surgery. Proceedings of Medicine Meets Virtual Reality 6 (MMVR-6), IOS Press, San Diego, California, pp 117-123.
- Bryan J, Stredney D, Wiet G, Sessanna D (2001) Virtual temporal bone dissection: a case study. IEEE Visualization 2001, October 21-October 26, San Diego.
- Cabral B, Cam N, Foran J (1994) Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. 1994 Symposium on Volume Visualization, pp 91-98.
- Cadoz C, Luciani A, Florens JL (1983) Responsive input devices and sound synthesis by simulation of instrumental mechanisms: the CORDIS system. Computer Music Journal, 8(3): 60-73.
- Chaigne A, Askenflet A (1992) On the use of finite differences for musical synthesis: application to plucked stringed instruments. Journal d'Acoustique 5(2): 181-211.
- Chaigne A, Askenflet A (1994a) Numerical simulations of piano string. I. A physical model for a stuck string using finite difference methods. Journal of the Acoustical Society of America 95(2): 1112-1118.
- Chaigne A, Askenflet A (1994b) Numerical simulations of piano string. II. Comparisons with measurements and systematic exploration of some hammer-string parameters. Journal of the Acoustical Society of America 95(3): 1631-1640.
- Choi BK, Jerard RB (1998) Sculptured surface machining theory and applications. Kluwer Academic Publishers, Norwell, MA.

- Chi X, Leu MC, Ochoa J (2004) Modeling of haptic rendering for virtual bone surgery. Proceedings of ASME International Mechanical Engineering Congress and R&D Expo and Computers and Information in Engineering Conference, Anaheim, CA.
- Chi X, Niu Q, Thakkar V, Leu MC (2005) Development of a bone drilling simulation system with force feedback. Proceedings of ASME International Mechanical Engineering Congress and Exposition, Orlando, FL.
- Chi X, Leu MC (2006) Interactive soft tissue deformation simulation using physics-based modeling. Proceedings of ASME International Mechanical Engineering Congress and Exposition, Chicago, IL.
- Chowning JM (1973) The synthesis of complex audio spectra by means of frequency modulation. *Journal of the Audio Engineering Society* 21(7): 526-534.
- Conditt M, Noble PC, Thompson MT, Ismaily SK, Moy G, Mathis KB (2003) Quantitative analysis of surgical technique in total knee replacement. Proc. of the 49th Annual Meeting of the Orthopedic Research Society, pp 13-17.
- Delp SL, Loan P, Basdogan C, Rosen JM (1997) Surgical simulation: an emerging technology for training in emergency medicine. *Presence* 6(2): 147-159.
- Depalle Ph, Helie T (1997) Extraction of spectral peak parameters using a short-time Fourier transform and no sidelobe windows. IEEE 1997 Workshop on Applications of Signal Processing to Audio and Acoustics, Mohonk, New York.
- Edmond CV, Heskamp D, Sluis D, Stredney D, Wiet GJ, Yagel R, Weghorst S, Oppenheimer P, Miller J, Levin M, Rosenberg L (1997) Simulation for ENT endoscopic surgical training. Proc. Medicine Meets Virtual Reality 5, San Diego, CA, pp 518-528.
- Epharim Y, Malah D (1984) Speech enhancement using a minimum mean-square error short-time spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-32 (6), 1109-1121.
- Epharim Y, Malah D (1985) Speech enhancement using a minimum mean-square error short-time spectral amplitude estimator. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-33(2), 443-445.
- Eriksson M, Flemmer H, Wikander J(2005) Haptic simulation of the milling process in temporal bone operations. MMVR 13 Medicine Meets Virtual Reality Conference, Jan 2005.
- Flanagan JL, Golden RM (1966) Phase vocoder. *The Bell System Technical Journal*, 45: 1493-1509.
- Florens JL, Cadoz C (1991) The physical model: modeling and simulating the instrumental universe. In: Poli et al. (eds), *Representations of Musical Signals*, The MIT Press, Cambridge, Massachusetts, USA, pp 227-268.

- Foley JD, Dam AV, Feiner SK, Hughes JF (1996) *Computer Graphics: Principles and Practice*, Second Edition. Boston: Addison Wesley.
- Friskien SF, Perry R. Simple and efficient traversal methods for quadtrees and octrees. *Journal of Graphics Tools* 2002; 7(3); 1-11.
- Funkhouser T, Jot J-M, Tsingos N (2002) Sounds good to me!—computational sound for graphics, virtual Reality, and interactive systems. SIGGRAPH Course Note.
- Galyean TA, Hughes JF (1991) Sculpting: an interactive volumetric modeling technique. *Computer Graphics* 4(25): 267-274.
- Gibson S, Samosky J, Mor A, Fyock C, Grimson E, Kanade T, Kikinis R, Lauer H, McKenzie N, Nakajima S, Ohkami H, Osborne R, Sawada A (1997) Simulating arthroscopic knee surgery using volumetric object representations, real-time volume rendering and haptic feedback. *Proceedings of Computer Vision and Virtual Reality in Medicine and Medical Robotics and Computer Assisted Surgery*, pp 369-378.
- Gladilin E (2003) *Biomechanical Modeling of soft tissue and facial expressions for Craniofacial Surgery Planning*. Ph. D. Dissertation, Free University, Berlin, GERMANY.
- Gorman PJ, Meier AH, Krummel TM (2000) Computer-Assisted Training and Learning in Surgery. *Computer Aided Surgery* 5:120-130.
- Haluck RS, Krummel TM (2000) Computers and virtual reality for surgical education in the 21st century. *Arch Surg* 135: 786-792.
- Hayes W, Bouxsein ML (1997) Biomechanics of cortical and trabecular bone: implications for assessment of fracture risk. *Basic Orthopedic Biomechanics*. Lippincott-Raven Publishers, pp 69-111.
- Held M, Klosowshi JT, Mitchell JSB (1995) Evaluation of collision detection methods for virtual reality fly-throughs. *The Seventh Canadian Conference on Computer Geometry*, 3: 205-210.
- Hiller L, Ruiz P (1971) Synthesizing musical sounds by solving the wave equation for vibrating objects: Part 1. *J. Audio Eng. Soc.* , 19(6): 462-470.
- Hobkirk J, Rusiniak K (1977) Investigation of variable factors in drilling bone. *Journal of Oral Surgery*, 35:968-973.
- Hua J, Qin H (2002) Haptic sculpting of volumetric implicit functions. *Proceedings of the 2002 IEEE symposium on volume visualization and graphics*, pp 55-64.
- Jackson A, John NW, Thacker NA, Gobbetti E, Zanetti G, Stone RJ, Linney AD, Alusi GH, Schwerdtner A (2002) “Developing a virtual reality environment for petrous bone surgery: a “state-of-the-art” review. *Journal of Otology and Neurotology*, March, 23: 111-121.

- Jacob CH, Berry JT, Pope MH, Hoaglund FT (1976) A study of the bone machining process-drilling. *J. Biomechanics*, 9: 343-349.
- John NW, Thacker N, Pokric M, Jackson A, Zanetti G, Gobbetti E, Giachetti A, Stone RJ, Campos J, Emmen A, Schwerdtner A, Neri E, Franseschini SS, Rubio F (2001) An integrated simulator for surgery of the petrous bone. *Proc Medicine mettes virtual reality*, pp 218-224.
- Karalis T, Galanos P (1982) Research on the mechanical impedance of human bone by a drilling test. *Journal of Biomechanics* 15(8): 561-581.
- Kaufman A, Cohen D, and Yagel R (1993) Volume graphics. *IEEE Computer* 26(7): 51-64.
- Kim L, Hwang Y, Park SH, Ha S (2005) Dental training system using multi-modal interface. *Computer-Aided Design & Applications*, 2(5): 591-598.
- Kovacevic D, Loncaric S, Sorantin E (1999) "Deformable contour based method for medical image segmentation. First Croatian Symposium on Computer Assisted Surgery, Zagreb, Croatia.
- Lacroute P, Levoy M (1994) Fast volume rendering using a shear-warp factorization of the viewing transformation. *Proceedings of SIGGRAPH '94*, pp 451- 458.
- Lang GF (1999) S&V geometry. *Sound and Vibration*, May, pp16-26.
- Levoy M (1988) Volume rendering - display of surfaces from a volume data. *IEEE Computer Graphics & Applications*, Los Alamitos, CA, 8(3): 29-37.
- Levoy M (1990) Efficient ray-tracing of volume data. *ACM Transactions on Graphics*, New York, 9(3): 245-261.
- Leu MC, Niu Q, Chi X (2007) Virtual bone surgery. *Virtual & Rapid Prototyping in Medicine*, SPRINGER-VERLAG, pp 21-44.
- Lin M, Gottschalk (1998) Collision detection between geometric models: a survey. In the *Proceedings of IMA Conference on Mathematics of Surfaces 1998*.
- Lorensen WE, Cline HE (1987) Marching cubes: a high resolution 3D surface construction algorithm. *Computer Graphics* 21(4): 163-169.
- Luis L, Schroeder W, Lydia N, Josh C (2003) *ITK software guide: the insight segmentation and registration toolkit*. Kitware Inc, USA.
- Maintz J, Viergever M (1998) A survey of medical image registration. *Medical Image Analysis* 2(1): 1-36.
- Mark WR, Randolph SC, Finch M, Verth JMV, Taylor II RM (1996) Adding force feedback to graphics systems: issues and solution. *Proceedings of the 23rd annual conference on computer graphics and interactive techniques*, pp 447-452.

- Massie TM, Salisbury JK (1994) The phantom haptic interface: A device for probing virtual objects. *ASME Haptic Interfaces for Virtual Environment and Teleoperator Systems* 1: 295-301.
- Meißner M, Huang J, Bartz D, Mueller K, Crawfis R (2000) A practical evaluation of four popular volume rendering algorithms. In *ACM Symposium on Volume Visualization*.
- McAulay RJ, Quatieri TF (1986) Speech analysis/synthesis based on a sinusoidal representation. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(6): 744-754.
- McNeely WA, Puterbaugh KD, Troy JJ (1999) Six degree-of-freedom haptic rendering using voxel sampling. *Proceedings of ACM SIGGRAPH*, pp 401-408.
- Miner NE, Caudell TP (2004) Method of sound synthesis. United States Patent, No. 678355 B1.
- Moorer JA (1985) Signal processing aspects of computer music: a survey. In: J. Strawn (ed.), *Digital Audio Signal Processing: an Anthology*, William Kauffmann, Inc., 5: 149-220.
- Morris D, Sewell C, Blevins N, Barbagli F (2004) A collaborative virtual environment for the simulation of temporal bone surgery. *Proceedings of Medical Image Computing and Computer-Assisted Intervention Conference, Saint-Malo, FRANCE*, pp 319-327.
- Morris D, Sewell C, Barbagli F, Blevins NH, Girod S, Salisbury K (2006) Visuo-haptic simulation of bone surgery for training and evaluation. *IEEE Computer Graphics and Applications*, 26(4):48-57.
- Nakao M, Kuroda T, Oyama H (2003) A haptic navigation system for supporting master-slave robotic surgery. *Proceedings of ICAT 2003, Tokyo, Japan*.
- Niu Q, Chi X, Leu MC (2005) Large medical data manipulation for bone surgery simulation. *Proceedings of ASME International Mechanical Engineering Congress and Exposition, Orlando, FL*.
- Niu Q, Leu MC (2007) Modeling and rendering for a virtual bone surgery system. *Proceedings of Medical Meets Virtual Reality Conference, Long Beach, CA*.
- Niu Q, Chi X, Leu MC, Ochoa J (2008) Image processing, geometric modeling and data management for development of a virtual bone surgery system. *Computer Aided Surgery* (in press).
- Okamura AM (1998) Literature survey of haptic rendering, collision detection, and object modeling.

- Pai DK, van den Doel K, James DL, Lang J, Lloyd JE, Richmond JL, Yau SH (2001) Scanning physical interaction behavior of 3D objects. *Computer Graphics (ACM SIGGRAPH 2001 Conference Proceedings)*, August 2001, pp 87-96.
- Peng X, Chi X, Ochoa J, Leu MC (2003) Bone surgery simulation with virtual reality. *Proceedings of ASME Design Engineering Computers and Information in Engineering Conferences*, Chicago, IL.
- Peng X, Zhang W, Asam S, Leu MC (2004) Surface Reconstruction from Dixel Data for Virtual Sculpting. *Proceedings of ASME International Mechanical Engineering Conference*, Anaheim, CA.
- Petersik A, Pflesser B, Tiede U, Hoehne KH, Leuwer R (2002) Haptic volume interaction with anatomic models at sub-voxel resolution. *Haptics 2002*, Orlando, Florida, pp 66-72.
- Pflesser B, Petersik A, Tiede U, Hohne HK, Leuwer R (2002) Volume cutting for virtual petrous bone surgery. *Comput. Aided Surg*, 7: 74-83.
- Plaskos C, Hodgeson A, Cinquin P (2003) Modeling and optimization of bone-cutting forces in orthopedic surgery. *Medical image computing and computer-Assisted Intervention - MICCAI 2878*;254-261.
- Ritter L, Burgielski Z, Hanssen N, Jansen T, Lievin M, Sader R, Zeilhofer HF, Keeve E (2004) 3D interactive segmentation of bone for computer-aided surgical planning. *Proceedings 4th Annual Conference of the International Society for Computer Assisted Orthopedic Surgery, CAOS'04*, Chicago, IL.
- Roads C (1995) *The computer music tutorial*, The MIT Press, Cambridge, Massachusetts, USA, pp 1234.
- Rodet C and Depalle P (1992) Spectral envelopes and inverse FFT synthesis. *Proceedings of the 93rd AES Convention*, San Francisco, California.
- Røtnes JS, Kaasa J, Westgaard G, Grimnes M, Ekeberg T (2002) A tutorial platform suitable for surgical simulator training (SimMentorTM). *Medicine Meets Virtual Reality 2002*.
- Ruspini DC, Kolarov K, Khatib O (1997) The haptic display of complex graphical environments *Proc of ACM SIGGRAPH*, pp 345-352.
- Ruspini D, Khatib O (1998) Dynamic models for haptic rendering systems. *Advances in Robot Kinematics: ARK98*, Strobl/Salzburg, Austria , pp 523–532.
- Schroeder W, Martin K, Lorensen B (2002) *The visualization toolkit – an object-oriented approach to 3D graphics*, Third Edition. New Jersey: Prentice Hall.
- Serra X (1989) A system for sound analysis/transformation/synthesis based on a deterministic plus stochastic decomposition. Ph.D. Dissertation, Stanford University.

- Serra X, Julius S III (1990) Spectral modeling synthesis: a sound analysis/synthesis system based on a deterministic plus stochastic decomposition. *Computer Music Journal* 14(4): 12-24.
- Shekhar R, Fayyad E, Yagel R, Cornhill J (1996) Octree based decimation of marching cubes surfaces. *Visualization* 96, pp 335-342.
- Shine NP, O'Sullivan PG, Connell J, Rulikowski P, Barrett J (2006) Digital spectral analysis of the drill-bone acoustic interface during temporal bone dissection: a qualitative cadaveric pilot study. *Otology & Neurotology*, 27: 728-733.
- Smith JO (1991) Viewpoints on the history of digital synthesis. *Proceedings of the International Computer Music Conference, Montreal, Canada*, pp 1-10.
- Smith, JO (1992) Physical modeling using digital waveguides. *Computer Music Journal* 16(4): 74-91.
- Strawn J (1980) Approximation and syntactic analysis of amplitude and frequency functions for digital sound synthesis. *Computer Music Journal* 4(3): 3-24.
- Sutton P, Hansen DC (1999) Isosurface extraction in time-varying fields using a temporal branch-on-need tree (T-BON). *IEEE Visualization '99*, pp 147-153.
- Takala T, Hahn J (1992) Sound rendering. *Proceedings of SIGGRAPH '92, ACM Computer Graphics* 26(2): 211-220.
- Tolonen T, Valimaki V, Karjalainen M (1998) Evaluation of modern sound synthesis methods. Helsinki University of Technology, Report 48, ISBN 951-22-4012-2.
- Udilijak T, Ciglar D, Mihoci K (2003) Influencing parameters in bone drilling. In *Proceedings of 9th International Scientific Conference on Production Engineering CIM*, pp 133-142.
- Valimaki V, Takala T (1996) Virtual musical instruments – natural sound using physical models. *Organized Sound* 1(2): 75-86.
- Van den Doel, Pai DK (1998) *The sounds of physical shapes*. Presence, the MIT Press, 7(4): 382-395.
- Vaseghi SV (2005) *Advanced digital signal processing and noise reduction* (third edition). John Wiley & Sons, Ltd., ENGLAND.
- Velasco F, Torres JC (2001) Cells octree: a new data structure for volume modeling and visualization. *Proceedings of the VI Fall Workshop on Vision, Modeling and Visualization, Stuttgart, Germany*, pp 151-158.
- Verma TS, Levine SN, Meng THY (1997) Transient modeling synthesis: a flexible analysis/synthesis tool for transient signals. *Proceedings of the International Computer Music Conference, Thessaloniki, Greece*, pp 164-167.
- VRMedLab, 2003, <http://www.bvis.uic.edu/vrml/> (last visted 10/06/2006).

- Wang SW, Kaufman AE (1995) Volume sculpting. Proceedings of Symposium on Interactive 3D Graphics, Monterey, CA, pp 151-156.
- Westermann R, Kobbelt L, Ertl T (1999) Real-time exploration of regular volume data by adaptive reconstruction of isosurfaces. *The Visual Computer*, 15(2):100-111.
- Westover L (1990) Footprint evaluation for volume rendering. SIG-GRAPH'90, pp 367-376.
- Widrow B, Glover JR Jr., McCool JM, Kaunitz J, Williams CS, Hearn RH, Zeidler JR, Eugene D Jr., Goodlin RC. (1975) Adaptive noise cancelling: principles and applications. *Proceedings of the IEEE*, 63: 1692-1716.
- Wiener N (1949) Extrapolation, interpolation, and smoothing of stationary time series, The MIT Press.
- Wiet GJ, Bryan J, Dodson E, Sessanna D, Stredney D, Schmalbrock P, Welling B (2000) Virtual temporal bone dissection. In: Westwood et. al. (Eds). *Proceedings of MMVR8*. IOS Press Amsterdam, pp 378-384.
- Wiet GJ, Stredney D, Sessanna D (2002) Virtual temporal bone dissection: an interactive surgical simulator. *Otolaryngol Head Neck Surg*, 127: 79-83.
- Wiggins KL, Malkin S (1976) Drilling of bone. *Journal of Biomechanics*, 9: 553-559.
- Wilhelms J, van Gelder A (1990) Octrees for faster isosurface generation. *Proceedings of the 1990 workshop on volume visualization*, pp. 57-62.
- Yang Z, Chen Y. (2003) Haptic rendering of milling. In *Proceedings of Eurohaptics Conference*.
- Zelinski R (1975) Noise reduction based on microphone array with LMS adaptive postfiltering. *Electronics Letters*, 26: 1692-1716.
- Zhu Q, Chen Y, Kaufman AE (1998) "Real-time biomechanically-based muscle volume deformation using FEM," *Proceedings of EUROGRAPHICS '98*.
- Zilles C, Salisbury JK (1995) A constraint-based god object method for haptics display. *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

VITA

Qiang Niu was born on February 2, 1971, in Mudanjiang, Heilongjiang Province, China. After attending high school in Mudanjiang, he studied at Northern Jiaotong University in Beijing. He received a Bachelor of Science degree in Mechanical Engineering in July 1993 and a Master of Science degree in Vehicle Engineering in July 1998 from Northern Jiaotong University. He received his second Master of Science degree in Computer Science from the University of Missouri-Rolla in May 2003.

From July 1993 to May 1999, Qiang Niu was employed by Northern Jiaotong University as an associate lecturer, and then as a lecturer. He taught two courses (Engineering Graphics and Computer Graphics I) and participated in many research projects sponsored by the China Railway Ministry.

Qiang Niu worked as a system analyst and software engineer at Beijing Bridge Electronics Ltd. in China from November 1998 to November 1999.

In June 2003 he began to pursue his Ph.D. degree in Mechanical Engineering at the University of Missouri-Rolla. During his graduate study at UMR, he held the position of Graduate Research Assistant in the Department of Mechanical and Aerospace Engineering. He graduated with a Doctor of Philosophy degree in Mechanical Engineering in May 2008.