INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

A VRML-Based Anatomical Visualization Tool For Medical Education

McGill University,
Montréal, Québec

February 11, 1997

A thesis submitted to the Faculty of Graduate Studies
and Research in partial fulfillment of the
requirements for the degree of

Master of Engineering

National Library of Canada
Acquisitions and Bibliographic Services
395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale du Canada
Acquisitions et services bibliographiques
395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre reference

Our file Notre reference

0-612-29635-0

Canada

ABSTRACT

The advent of the Virtual Reality Modeling Language (VRML) as a portable file format for describing three-dimensional (3-D) scenes has created the ability for researchers, educators and students to share anatomical models on the WWW. The implication for medical teaching is that students can interactively examine anatomical structures and their 3-D spatial relationships, using current personal computer (PC) technology. The work of this thesis creates, for the first time, a high-resolution middle-ear model that is accessible on the World Wide Web (WWW). The 3-D model is created by automated aligning of the source images (histological sections), interactive segmentation, and 3-D surface reconstruction. The resulting model is translated into VRML format. Images of the histological sections can be superimposed on the model, allowing students to view a section in its 3-D context. To enhance the viewing of these scenes, a VRML browser is modified to support transparent rendering of surfaces. Finally, a WWW interface is designed to allow users to selectively choose the model structures, section images and associated viewing parameters, in order to build their own 3-D scene.

RÉSUMÉ

L'arrivée du language de Modelisation de la Réalité Virtuelle (*Virtual Reality Modeling Language,* VRML), comme format de fichiers portatif pour décrire des scènes tri-dimensionnelles (3-D), a permis aux chercheurs(es), aux éducateurs(rices), et aux étudiants(es) de partager des modèles anatomiques sur le W3. L'implication dans le domaine des études médicales est le suivante: les étudiants(es) peuvent examiner d'une manière interactive des structures anatomiques et leurs associations spatiales en 3-D, avec l'aide d'ordinateurs personnels (PC) de technologies actuelles. Le travail de cette thèse consiste à réaliser, pour la première fois, un modèle de l'oreille moyenne de haute définition qui soit accessible sur le *World-Wide-Web* (W3). Le modèle 3-D de l'oreille moyenne est créé par l'alignement automatisé des images d'origine (des sections histologiques), la segmentation interactive et la reconstruction des surfaces 3-D. Le modèle final est traduit en format VRML. Des images des sections histologiques peuvent être superposées sur le modèle, permettant aux étudiants(es) de voir une section dans son contexte spatial tri-dimensionnel. Pour accroître la perception des images, un navigateur VRML a été modifié pour permettre des rendus de surfaces transparentes. Finalement, un interface W3 a été conçu pour permettre aux utilisateurs de choisir les structures du modèle, les images des sections et les paramètres de visualisation associés, nécessaires à la construction de leur propre scène 3-D.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I wish to thank my advisor, Professor W. Robert J. Funnell, for the many discussions on a moment's notice, thoughtful direction when I needed it, and his constant challenge to strive for clarity in my writing; my wife Sylvaine, for her support and love throughout my studies, my little daughter Nolwenn for teaching me that there is more to life than abstractions, and my brother Paul Warrick for providing a medical student's perspective in his proof-reading of the text; Zsuzsanna Bencsáth-Makkai for her careful reading of the text; Bruce Davies at the Montréal Neurological Institute (MNI) for his helpful introduction to the details of 3-D reconstruction; Keith Andrews, Michael Pichler, and Georg Mészáros of the VRweb project for advice and cooperation in extending the VRweb VRML browser; David Macdonald at the Brain Imaging Center for providing and debugging his *display* program for my environment; and finally Shelly Feran at the Medical Computing Resource for her system help with compilers, image tools, and other details.

This work was supported by the Medical Research Council of Canada and McGill University.

1. INTRODUCTION

A transformation in the methods and materials used to acquire, organize, and distribute medical knowledge is underway. This will have particular impact on the medical education process, as an ever-increasing body of knowledge must be encountered and assessed by both undergraduate medical students and experienced medical practitioners.

Some have postulated that it is information technology that will facilitate mastery over the growing body of medical knowledge [Barnett 1989]. Information technology is now being routinely used in medicine to help cope with these pressures. Indeed, the appearance of this technology in medicine is evidence of the increasing importance of certain biomedical research disciplines. *Medical informatics*, *visualization*, and *virtual reality* are three inter-related domains that present novel possibilities for conveying and understanding medical knowledge, and in particular anatomically related knowledge. Definitions of these key terms and explanations of their relevance to medical education will establish a framework wherein the work of this thesis is situated.

1.1 Medical Informatics

Medical informatics has been defined as "the rapidly developing scientific field that deals with the storage, retrieval, and optimal use of biomedical information, data, and knowledge for problem solving and decision making" [Shortliffe and Perreault 1990]. It is important to emphasize that the methods of medical informatics play a subservient role in achieving medical goals:

> "[it is] a developing body of knowledge and set of techniques concerning the organization and management of information in support of medical research, education, and patient care" [Jennett *et al.* 1991].

Gupta feels that the ability to perform the components of the medical task (diagnosis, decision making, management, and prognosis) is dependent upon the physician's ability to continually assimilate new and updated information, pointing to a growing need for a

paradigm shift in medical education [Gupta *et al.* 1995]. This was recognized at least a decade ago when, in 1986, the Association of American Medical Colleges (AAMC) convened both medical and informatics leaders and formulated numerous recommendations, including the proposal that medical informatics should become an integral part of the medical curriculum [AAMC 1986]. Much has been attempted since then to respond to this proposal, but much more research and experience is required to determine how and when to integrate the appropriate technologies with traditional methods.

1.2 Visualization

Simply stated, visualization is the transformation of data or information into pictures. It assumes a cooperative role with an active human subject:

> visualization engages the primary human sensory apparatus, vision, as well as the processing power of the human mind. The result is a simple and effective medium for communicating complex and/or voluminous information [Schroeder *et al.* 1996].

Scientific visualization is a term that originated from the 1987 National Science Foundation (NSF) report *Visualization in Scientific Computing*. "It is the computer science field that encompasses user interface issues, data representation and processing algorithms, visual representations, and other sensory presentation such as sound or touch" [McCormick *et al.* 1987].

Rosse argues that anatomy, among all biomedical disciplines, has the most to benefit from computer three-dimensional (3-D) visualization and knowledge representation methods because they can present data in its full 3-D form and at the same time can convey organizing notions of classification and relation [Rosse 1995]. Addition of 3-D information will create new, more effective ways of assimilating anatomy and histology. Interactive, true 3-D atlases of anatomy are now being created, which allow dissection-like exploration from any viewing angle. In addition, they also provide explanations of the anatomical concepts that traditionally have been found in anatomy texts. Similarly, examination of

microstructures can be united with the concepts of histology, assuming adequate data resolution exists.

1.3 Virtual reality

The definition of scientific visualization existed well before the term virtual reality (VR) had the commonplace usage it has today. The two are certainly interrelated concepts. VR concerns itself with the creation of synthetic environments that approximate real environments, both in their presentation to human perception, and their response to human input. VR systems can generate computer-based models of the real world and can provide humans with the means to interact with these models through new human-computer interfaces. In doing so, the objective is to approach realism in the experience of the models [Kaltenborn and Rienhoff 1993]. Although there is no full agreement as to exactly what constitutes a VR system, they all tend to possess these characteristics to varying degrees:

1) *3-D representations* of the VR model by the computer: auditory and force-feedback (haptic) information can be added to the visual representation in order to make the representation of the virtual world more realistic. Special equipment, such as head-mounted displays that incorporate stereo vision, can be used to realize these demands.

2) *Real-time interactivity:* our visual system requires scene updates on the order of 20-30 Hz for *fusion* to occur; that is, when the consecutive discrete images present the illusion of continuous motion. In addition, significant system lags in response to user input can be extremely disorienting to the human sensory apparatus, to the point of inducing motion sickness.

3) *Ergonomic interfaces*: a human-oriented interface design which facilitates natural modes of communication, such as gestures and language, rather than the typical computer-oriented interfaces such as keyboard and mouse. VR technologies offer special human-centred interfaces, such as data gloves and body suits.

Maximizing the degree of *immersion* of a VR system is also considered crucial to the realism of the environment. The perception of immersion is increased when all the above characteristics have been successfully integrated into the interactive, synthetic world: "studying images on a 2-D computer screen can be compared to looking at fish through a glass bottom boat, whereas VR allows one to put on the scuba equipment and enter the water, interacting with the surroundings without getting wet" [Sprague *et al.* 1993].

Currently, the majority of medical interest in VR is driven by the needs of surgical education, diagnosis and treatment. Training of surgeons is an expensive process: often a minimum of 30-50 supervised surgical procedures are required for a trainee to gain certification to perform a solo procedure. In addition to the high cost of this process, mistakes are made at a disproportionate rate by surgical residents during their first surgical undertakings on live patients [Johnson 1996]. These factors have motivated numerous researchers to develop surgical simulators that allow the resident to practice surgery on virtual patients before entering the operating room. This research and other related activities will be providing the basis for immersive anatomical education tools in the near future.

1.4 Outline of our work

Chapter 2 presents a selected review of the literature pertaining to computer applications to medical education, anatomical visualization in 3-D, and virtual reality in medical training.

Subsequently, we describe our implementation of interactive visualization techniques. The focus of the project is on creating and displaying 3-D models. The processes used to create 3-D anatomical models of cat middle-ear structures from histological sections are described in Chapter 3 and 4. A method of describing these models and their corresponding 3-D histological sections using the Virtual Reality Modeling Language (VRML) is presented in Chapter 5. The goal of this development is to create tools which can provide a better understanding of where the sections are situated in three-dimensional space, and how they relate to the 3-D structures of which they are a part. To enhance viewing of the VRML models, transparent rendering is added to a VRML browser: in Chapter 6, we discuss the computer-graphics fundamentals involved, and present an implementation of interpolated

transparency. In order that the tools and data be widely and readily available to medical students, the target platforms include current PC technology and the World-Wide-Web (WWW). Chapter 7 describes how custom VRML 3-D scenes can be constructed and requested by the user on the WWW. This facility can be found at the following URL address:

`http://funsan.biomed.mcgill.ca/cgi-bin/phil-cgi.pl`

Chapter 8 evaluates the project in terms of its usefulness and limitations, and proposes future areas of research in order to create more comprehensive instructional tools of this type for anatomical education.

2. VISUALIZATION IN MEDICAL EDUCATION

2.1 Introduction

This chapter will present a brief survey of the literature on the use of visualization methods in medical education, with special emphasis on anatomical training. First, the place of these approaches will be examined within the domain of computer-based teaching in medical education. Then the role of 3-D visualization and virtual reality in medical teaching will be addressed.

2.2 Computer applications in medical training

2.2.1 Educational datasets

While technology is finding an increasing role in the medical domain in general, there are certain distinctive features of endeavours having an educational rather than clinical focus. The use of educational datasets, restricted but representative sets of data, can be sufficient to demonstrate many concepts. In clinical settings, however, individual patient data are the basis of diagnostic decisions: given the current state of hardware and software technology, such patient-specific processing often requires unreasonable time and effort. This lengthy and often tedious data preparation may be justified for single datasets used for education purposes. It must be recognized, of course, that such data selectivity necessarily restricts a student to a small set of representative cases that may not consider variation and pathology. Another feature of working in the educational domain is that source data need not be human (assuming animal specimens are available) if the inter-species variation is low enough to demonstrate a particular concept.

2.2.2 Hypermedia

Currently, most applications of computers in medical curricula have concentrated on two-dimensional (2-D) hypermedia approaches to integrate text and images. [Stanford *et al.* 1994, Dev *et al.* 1992, Alper *et al.* 1991, Pradhan and Dev 1993] These have played a

useful role in providing a more efficient method of access to an increasingly large body of information [Schubert *et al.* 1994], and have been especially effective when used as an adjunct to traditional approaches such as cadaver dissection. Stanford *et al.* determined, for example, that a hypertext multimedia program for teaching cardiac anatomy improved performance on image identification tests, when combined with cadaver dissection, as compared with the use of dissection alone [Stanford *et al.* 1994].

2.3 Learning issues

One principle that underlies much of the shift to incorporating new technologies into the medical curriculum is the following reflection:

> People retain only 20% of what they see and 30% of what they hear. But they remember 50% of what the see and hear, and as much as 80% of what they see, hear, and do simultaneously [cited in Gupta 1995].

Few in the educational-technology field would disagree with the premise that increasing the interaction of a learning activity further engages mental processes and improves motivation [Jonassen 1985, Jih and Reeves 1992]. While there may be dispute over the details of the cited quantitative formulation, it seems clear that medical teaching has begun to recognize the advantages of technology-based interactive learning.

Certain recognized learning techniques should influence, or be provided for, in pedagogically sound computerized medical teaching. First of all, in order for these tools to be accessible and easy to learn, a simple, intuitive user interface that allows efficient exploration is required. If students cannot understand how to use the basic functionality of the software, the learning potential is obviously diminished. This is especially important for users who are unfamiliar with computers. While the graphical interfaces and presentation tools available on most current platforms have greatly increased the ease of use of computers in general, there are decisions specific to each application that determine their effectiveness in accessing information in an intuitive fashion.

There also needs to be assistance to allow the user to be guided through the data in a methodical manner. Given the increasing sizes of the medical datasets, it is evident that unstructured browsing through a huge volume of data can be counterproductive. There must be help for the student to know when to quit, satisfied that he or she has gained the knowledge that was sought [Eno *et al.* 1992].

A certain degree of repetition and reinforcement is an important part of the learning process. Quizzes and drills enable student self-assessment, and under the proper conditions can also be used by the educator for academic evaluation. In the context of teaching histology or anatomy, for example, these drills can take several forms. The student can be asked to name every structure in an image whose component structures have been clearly delineated (outlined). A set of such images can be displayed in random order. If desired, presenting multiple-choice responses can reduce the difficulty for the student if the number of choices is small. Another approach is to have a very large number of choices (perhaps the entries of an entire medical dictionary), to reduce the syntactical and semantic problems of free-text user input, such as spelling and synonyms. The identification can be reversed with a "pointing quiz": a structure name is displayed to the user, who is asked to point to the corresponding structure on the computer screen. In this case the degree of difficulty can be modified by displaying or hiding the structure outlining or colour segmentation [Eno 1992].

Alper's system HistoLogical [Alper 1991] focuses on having feedback for different erroneous responses, in order to impart a tutor-like quality to the software. For example, if a student is shown a histological section of cardiac muscle, and the student identifies it as skeletal muscle, the program judges it as a critical mistake and displays the two types simultaneously for comparison.

Incorporating this type of intelligence into learning software helps the student to establish valid and flexible *conceptual structures* [Pradhan and Dev 1993]. This term has been used to describe the mental models of a domain that are formed and refined as students are presented with complex and multi-faceted material. Misconceptions can arise when excessive demands are made on the student's cognitive abilities, causing the conceptual

structures to break down. These misconceptions can take several forms. Oversimplification of complex anatomical structures, for example, does not account for the wide variability of structures within and between normal and pathologic subjects. Over-reliance on a single basis for mental representation can cause problems when a new case does not fit the conceptual framework. Mental "recipes" that no longer consider the logic of a sequence of steps can become entrenched. Any such attempts to rigidly compartmentalize knowledge components make it difficult to respond to unique or pathological cases. What is needed is to construct sound reasoning principles that promote adaptation to new circumstances. In accordance with Alper's approach, Pradham suggests that computer-based instruction should target these most common misconceptions.

2.4 Visualizing anatomy in 3-D

2.4.1 Importance to anatomy education

Rosse is a strong proponent of introducing 3-D data into the medical anatomy curriculum in order to provide better conceptual frameworks for clinical diagnosis. His view is that

> the ultimate goal of anatomical training is to assist the student in gaining a full understanding of the 3-D dynamic structure of the living body, in order to apply the appropriate cognitive skills when clinical problems require anatomical reasoning.

Anatomical reasoning is a fundamental tool for clinical decision-making:

> [it is] the cognitive process that relates manifestations of normal and abnormal function to anatomic entities and seeks to explain these manifestations in terms of the attributes of different anatomic structures [Rosse 1995].

The importance of possessing a three-dimensional understanding of anatomy when confronted with unforeseen conditions is emphasized by the findings of Pradhan and Dev. Their evaluation of traditional medical education is that it has concentrated on teaching facts rather than concepts. They studied the problem-solving skills of students who had used a software program depicting 2-D images of neuroanatomy. The program could be used to identify structures on section images, and provided a means of self-evaluation through

quizzes. A subsequent evaluation required the students to identify structures on 2-D images and to respond to scenarios requiring higher-level problem-solving skills. All students used the software, but those who had acquired a 3-D conception of the anatomy performed better on the evaluation than those who remembered lists of facts. Importantly, this was especially true when the presented images deviated from the norm. Many students were limited in the information they could retrieve by the organization and order of the information they memorized [Pradhan and Dev 1993]. These findings suggest that a physician's ability to accommodate new information is enhanced by 3-D notions of anatomy.

Promoting anatomical reasoning with computerized representations of anatomy must address two domains: what Rosse refers to as the spatial (entities or their images) and symbolic (descriptions of concepts and relationships). Traditionally, atlases of 2-D images and cadaver dissection have presented spatial information while anatomy textbooks have articulated symbolic concepts. Recent 2-D hypertext-multimedia approaches have integrated these two sources of information. Some researchers in 3-D anatomy, however, have referred to them as improved access to the same old restricted data [Schubert *et al.* 1994]. Rosse contends that computer-based knowledge sources will not manage the information load unless they introduce qualitative differences in presentation and organization.

Rosse further argues that anatomy, among all biomedical disciplines, has the most to benefit from computer 3-D visualization and knowledge-representation methods, because they allow convergence of the two anatomical domains. They can present data in its full 3-D form and have the potential to convey classifications (of structures with common characteristics) and relationships (to other biomedical concepts). A description of two projects that attempt to bridge the two domains will follow in section 2.5.

2.4.2 Comparison with traditional methods

The introduction of new teaching methods in anatomy necessarily calls into question the role of traditional methods. Cadaver dissection has long been considered an essential element of anatomical training, but given the high costs involved in maintaining a will-

donor program, preparing the specimens and adhering to environmental safety requirements [Rosse 1995], some have questioned its essential role. While conceding that cadaver dissection has many important advantages (requiring teamwork, time for debate and first-hand exploration), Rosse feels that 3-D computer techniques offer several important advantages:

> Even without displaying the models in 3-D space, and without the tactile characteristics of anatomic structures, currently available computer-graphics techniques afford a richer appreciation of the three-dimensional quality of anatomy than is possible with a cadaver. Not being destroyed by their disassembly, as is the cadaver by dissection, segmented spatial models lend themselves to repeated explorations that may be guided by different objectives. For instance, the models can provide a broad view, appropriate for 3-D orientation, or focus on the details of structures and spatial relationships, appropriate for the planning of surgical procedures. Two capabilities of segmented computerized models particularly enhance the generation of a cognitive model by the trainee: (1) they can display anatomy not only from a regional point of view, as does cadaver dissection, but also by organ systems; and (2) this display can be generated not only by electronically taking the model apart ("dissecting it") but by building up the body systems or regions from their components [Rosse 1995].

This shift of emphasis may lead to questions such as "what can the student learn in dissection that cannot be learned with the model?"

Although some medical schools have reduced their dissection programs to the display of prepared specimens (eg., McMaster University) or abandoned their dissection programs altogether (eg., Université de Montréal), most of the medical education community remains rather more cautious. While admitting the possible merits of certain new technologies for the medical-school curriculum, one medical-informatics researcher cautions that they have not been scientifically proven to attain a level of pedagogical quality that surpasses (or even equals) traditional methods [Gupta *et al.* 1995]. Rather than sacrificing any class time, they feel, new educational methods should initially be used exclusively as an adjunct to traditional teaching. Furthermore, all new technologies should undergo an "educational benefit analysis", comparing the performance of a control group without the technology to a test group with access to the technology. (They add that all parties should be informed about

the analysis beforehand. It should be mentioned, however, that it is difficult to have cooperation in this type of controlled study, where a subset of the student population has their access to learning materials arbitrarily restricted.)

It is nevertheless important to emphasize that both dissection and 3-D models present advantages and disadvantages, and that a complementary approach may be most beneficial in the short-term. Models with tactile feedback are still in the preliminary phases of research. In response to one of Rosse's points, certainly valid, that repeatable non-destructive exploration is advantageous, it should be recognized that tissue and organ destruction may be important to experience and appreciate. An anatomy program will be most successful if it integrates both approaches and considers wisely the use of limited resources. 3-D models, for example, could be used for the initial learning phase, preparing a well-informed student for a dissection guided by clear objectives. In any case, if 3-D models are to be used, complete integration into the curriculum is essential, since student participation tends to be low for optional learning activities.

2.4.3 Sources of spatial data

The main sources of spatial data are MRI, CT, ultrasound, and histological sections. The educational usefulness of these data is dependent on several criteria, including signal-to-noise ratio, coverage, resolution and degree of segmentation. If the data can encompass the whole body, a complete representation is possible for all anatomical regions. If the data resolution is fine enough, the ability to visualize smaller structures will be enhanced. If the data is segmented, it is much easier to emphasize individual entities and understand their structural context.

It is important to have access to multiple data sources since they do not represent all aspects of anatomy equally well. Some tissues have greater contrast in certain modalities than others. CT, for example, is very effective for detecting bone interfaces, while MRI is particularly suited to soft tissue such as muscle. Histological sections obtained from cadaver specimens (using cryotomy or an embedding material such as plastic, celloidin, or paraffin) generally give the best resolution but, in addition to the obvious restriction to cadaver

specimens, suffer from problems of distortion, misalignment and damage during preparation. The Visible Human Project of the National Library of Medicine [Ackerman 1992] has made available datasets of male and female subjects from three modalities. The resolutions of some of these data (see Table 1) are just beyond what is currently available with standard clinical techniques.

Data Source	MRI	CT	Histological
Male	1.0×1.0×4.0	0.53×0.53×1.0	0.33×0.33×1.0
Female	same as male	same as male	0.33×0.33×0.33

Table 1: Resolutions of the Visible Human dataset (units in *mm*).

The raw data are much more useful when they has been segmented. For all structures of interest, segmentation involves identifying their corresponding regions and delineating their extents. Segmentation is still predominately a manual, labour-intensive process, although automatic or semi-automatic algorithms for segmentation is a subject of much ongoing research [Medine *et al.* 1995, Vinitski *et al.* 1995, Warscotte *et al.* 1995, Höhne *et al.* 1992]. Currently, such processing may not be feasible for routine clinical care, but is quite realistic for a reusable teaching tool.

Techniques for graphical display of the segmented data fall generally into two broad categories: surface rendering and volume rendering. Surface rendering refers to the display of surfaces from iso-contours within a volumetric data set. For example, when trying to visualize a tumour one might use tracing or edge detection to isolate the edges that represent the tumour in each 2-D slice and then use surface-reconstruction techniques in order to connect the edges in adjacent slices to produce a 3-D surface. The resulting polygonal surfaces are well suited to computer graphics displays. Volume rendering refers to the rendering of a 3-D distribution in order to show the characteristics of any solid region. For example, a volume rendering of a human head not only allows the user to view the surface of the skull or brain, but also allows viewing interior regions, using techniques such as

transparency or plane cutting. The choice between surface and volume rendering is ultimately dictated by the needs of the application as well as by the available hardware: volume rendering can be extremely computationally demanding, owing to the order n^3 (written as $O(n^3)$ henceforth) nature of processing volumes.

2.4.4 Sources of symbolic data

Creation of sources of symbolic anatomical information has begun: one such initiative is the Unified Medical Language Systems (UMLS) project of the National Library of Medicine [Lindberg *et al.* 1993]. The goals of this work include:

> formalizing descriptions for comprehensive sets of anatomical concepts, the terms that denote them, and the linkages that represent hierarchical and other relationships among anatomical concepts [Rosse 1995].

The linkage from spatial anatomical entities to attributes can occur for a wide variety of anatomical concepts (functional, developmental, and topological relationships would be a few examples), and thus conventional database design (relational or object-oriented) can be an appropriate organizing construct. In addition, however, anatomical knowledge has features that cannot be characterized by such linear techniques. To address non-deterministic relationships—where the relation is 'maybe', 'sometimes' or 'most of the time' true—tools of artificial intelligence such as semantic networks, conceptual graphs, belief networks and rule-based formalisms may be more suitable. Once developed, these schemes will add intelligence to the spatial models.

2.5 Integrating anatomical spatial and symbolic domains

2.5.1 VoxelMan project [Schubert et al. 1994]

Researchers at the Institute of Mathematics and Computer Science in Medicine (Hamburg) believe that full volume 3-D visualization, rather than the hollow shell reconstructions of surface renderings, is the route to realistic simulations of dissection. Most atlases give just a few views of a region, but the ability to perform arbitrary cuts greatly enhances visualization (especially for surgical planning). Such ability requires access to complete volume data. The VoxelMan tool is such a complete space-filling model, allowing multiple surface display,

transparent material, and arbitrary cutting planes. Hollow structures such as the brain ventricles can be displayed as solid objects. Simulated x-rays (tomograms) from CT projections are possible by summing up the voxel absorption values along a viewing ray. It is also possible to mix both sectional images and 3-D objects to aid radiological training.

In addition to these spatial domain functions, this project integrates an anatomical knowledge base to directly link structures with concepts. The spatial organization underlies all aspects of the system. Each voxel is classified based on its membership within various regions. (Voxels at structure boundaries possessing multiple materials—due to *partial volume* effects—may not be included in any region. The degree of this inaccuracy is resolution dependent.) Many kinds of regions are conceivable; the VoxelMan implementation has included the following:

Region	**Example**
structural regions	the precentral gyrus of the brain
functional regions	the sensorimotor cortex
region blood supplier	the anterior cerebral artery
pathological regions	regions involved in a certain pathology
histological regions	regions belonging to a histological class of tissue

Table 2: Examples of VoxelMan regional classifications

At the higher level of symbolic organization is a knowledge base containing the hierarchical relations between basic regions:

Relationship	Example
structural relations	the precentral gyrus is part of the frontal lobe
functional relations	stimuli associated with pain, temperature, or light touch travel the spinothalamic tract
topological relations	the internal carotid artery supplies the anterior cerebral artery and the middle cerebral artery
	the thalamus is lateral to the third ventricle

Table 3: Examples of VoxelMan hierarchical relationships

Not surprisingly, the procedure for identification and classification is long, pain-staking work: a neuroanatomist required the equivalent of three months of full-time work to fill the hierarchical structure with 300 objects.

However, with the knowledge in place, any voxel can be clicked to query the database. In this way questions like: "What am I pointing to?", "Where is it?", "What is it part of?", "What is it connected to?", "What does it do?", and "What are its neighbouring structures?" can be answered. The inverse sort of query can also be achieved: an object can be selected from the knowledge base in order to have its image(s) appear.

This project first created results in 1990 using MRI and CT data, even though the resolution was inadequate. Comprehensive histological sections were not available until 1994 when the Visible Human Project data was first released. They have now incorporated this data as well [Tiede *et al.* 1996].

2.5.2 Digital Anatomist [Eno et al. 1992]

This project of the Department of Biological Structure at the University of Washington takes a multimedia approach to organizing anatomical data. Rather than focusing on interactive 3-D, the Digital Anatomist integrates 2-D images, movies of 3-D renderings of segmented data, and text managed by several databases. It is a WWW-based information

source: clients can always have access to the most up-to-date data, and information can be stored in a distributed fashion. Collaborative work is not only possible, it is unavoidable: their view is that the amount of data is large enough to overwhelm the human and computing resources of individual research groups.

The Digital Anatomist is organized by four system databases:

1) a *spatial database* consisting of 2-D image slices, segmented slices, and pre-computed 3-D reconstructions.

2) a *symbolic database* containing the information necessary for retrieving objects from the spatial database (eg., the filenames for images of the lateral ventricles of the brain).

3) a *symbolic knowledge base* consisting of well-defined nomenclature and classification for each symbol in the symbolic database:
   a) synonyms: e.g., ventriculus lateralis is a *synonym* of lateral ventricle.
   b) symbolic relationships: eg., lateral ventricle *is-a* ventricle, lateral ventricle is a *subdivision* of the telencephalon, the anterior horn of the lateral ventricle is a *subdivision* of the lateral ventricle.
   c) other textual attributes: eg., a term for a glossary: "the lateral ventricles are a pair of large irregular shaped cavities which contain cerebrospinal fluid and are located in the cerebral hemispheres..."

4) a *spatial knowledge base* describing shape and range of variation of structural objects. e.g., the average shape and range of variation of the lateral ventricles (inter-subject variation) and the geometric relationships between the ventricles and other brain parts: where it is situated with respect to other sections of the brain.

In order to query these databases, a WWW browser has been created for use on the Macintosh platform. All computationally intensive processing (such as 3-D reconstruction) is done on the server machines. With this design and functionality, the Digital Anatomist does not place heavy demands on the client machine.

2.6 Virtual reality in medical training

An increasing amount of research is being devoted to the application of virtual-reality technology to visualization in medical training. So far, virtual-reality displays have typically been limited to simple, stylized models, but this will change as hardware capabilities improve [Schubert *et al.* 1994]. The restriction to limited endeavours has been due to the large processing requirements for volume-rendering approaches, and the largely open problems of segmentation and classification for the delineation of structures [Meinzer 1994]. The degree of interaction and immersion has not yet fully matured either, but some environments have been developed that focus on a few of these goals. Their application includes training for surgical and medical procedures, and general anatomical education.

Surgical simulation and pre-operative or intra-operative visualization have been seen as potential applications of virtual reality. Endoscopy and minimally invasive surgery (MIS), for example, currently rely on 2-D video images of 3-D anatomical space for navigation, which leads to errors in eye-hand coordination and spatial orientation [Kaltenborn and Rienhoff 1993]. It is hoped that such surgery can be improved using 3-D displays and tactile feedback. Such image-based procedures are likely to be the first to benefit from virtual reality: full-blown surgical simulations complete with tissues that feel realistic to the touch (i.e., using physically based models) and that behave as expected (e.g., bleeding when cut) are a much more difficult problem, and may take several more decades of research [Higgins 1996]. In the meantime, hybrid simulators now exist that use mannequins for physical procedures (such as endoscopic insertion), in conjunction with other computer simulations.

Some have suggested that virtual reality can be used for accreditation of surgeons: if a surgeon does not fall within a specified standard deviation of the the norm for performance of a particular procedure, they cannot perform that surgery. This testing, it is argued, is already in place for other professions, such as pilots, whose psychomotor, perceptual and cognitive skills are responsible for human lives, so surgical performance should require comparable scrutiny [Johnston 1996].

Virtual reality is now applicable to standard medical procedures such as IV- needle insertion. Nurses can train in an immersive environment (i.e., wearing VR goggles) and sense the force feedback on their instrument: they feel resistance and pops as they puncture first the skin surface, and then the vein [Higgins 1996].

Merril *et al.* describe a similar system to simulate the retrobulbar injection procedure, which is used during ophthalmic surgery as an anesthetic for pain control, as well as to paralyze ocular motility [Merril *et al.* 1992]. This is a delicate task, and prone to complications if incorrectly administered. Thus the targeted user group for this educational tool is ophthalmology residents. Their system consists of a model head, a position-sensed probe, and a 3-D reconstruction of the orbital structures. The position of the probe is continuously updated on the 3-D scene, indicating points of entry into the various layers of tissue. No mention is made of a force-feedback mechanism in their system description, nor of how the probe interacts with the model head. In addition to the motor-skill aspect of this system, other multimedia elements are incorporated, including a series of annotated images of human eye sections, and digitized movies of the procedure in the operating room.

This type of application to anatomical education is relatively new. Another project that attempts to integrate a virtual-reality environment with existing multimedia information sources is the Virtual Reality-Multimedia Synthesis project [Hoffman 1995, Hoffman *et al.* 1995]. These researchers have integrated elements of virtual reality (3-D scenes of anatomy, with the eventual goal of surgical simulation), multimedia (normal/abnormal anatomy images and explanatory text), and communications (access to Internet resources and local hospital patient records) into a system that they hope to use within their medical curriculum. The first lessons they have developed include 3-D models of the hepatobiliary system, which provide the core anatomical content to which multimedia links are supplied. Other technical details of this project have not yet been clearly described in their articles. Rather than focusing on technology per se, they prefer to place special emphasis on instructional design and end-user need (e.g., adaptation to the skill level of the user). In

consultation with other members of their medical faculty, their priority is to respond to demonstrated educational need rather than to establish completely new paradigms.

2.7 Conclusions

It should be clear that visualization methods move beyond mere image display for conveying information: they can incorporate other existing and extensive resources and knowledge bases. This is creating a growing interest in and need for structured knowledge representations to support such information loads. As the same time, the integration of other media and stimulation of other senses beyond the visual are becoming increasingly important for comprehensive visualization environments. Perhaps most importantly, it is becoming recognized that implementations must be based on sound pedagogical techniques in order to be useful and accepted.

3. VIEWING HISTOLOGICAL SECTIONS IN THEIR 3-D CONTEXT

3.1 Scope of our work

The motivation for the work of this thesis is drawn from a dichotomy that now exists within anatomical visualization. On the one hand, there are 2-D hypermedia applications displaying predominately images and text, which can function adequately on current personal computer platforms. On the other hand, there are sophisticated environments displaying segmented 3-D models in an interactive environment, with arbitrary slicing ("virtual dissection"), surface transparency, lighting, etc., which are accessible only to high-end workstations, especially if reasonable interactive rates are desired.

We have attempted to strike a middle ground between these approaches. 3-D surface models of the cat middle ear, reconstructed from histological sections, are simultaneously displayed with their correctly positioned section images. In this way it is hoped that the 3-D context of the section image can be more fully appreciated by the user.

The importance for medical practitioners to possess a 3-D anatomical framework that is sound and flexible has already been underscored in the previous chapter. Given that this is the case, it has been hypothesized that the ability to localize a cross-section in the body is a measure of one's anatomical expertise [Dev *et al.* 1992]. If there is some validity to this observation, it gives strong motivation for our work: by placing histological sections in the context of 3-D structures, it is hoped to facilitate development of such localization skills, and in doing so, to promote *anatomical reasoning*.

In order for such a tool to be useful, it must be accessible. For this reason, we have chosen to integrate the tool with existing World-Wide-Web (WWW) resources. The Virtual Reality Modeling Language (VRML) is a Web format that allows polygonal 3-D models to be shared on the Internet. We use VRML to describe and distribute our 3-D middle-ear anatomy, so that it can be viewed by medical users using publicly available VRML

browsers. We have modified one browser in particular to support transparent rendering, in order to have an additional viewing mode that presents slices and 3-D models simultaneously. In addition, the 3-D scenes can be interactively "built" using a WWW browser (such as Mosaic or Netscape), to allow selection of scene parameters such as slice number, colours and transparency values.

The focus of our work is on creating and viewing visualizations of spatial data of the middle ear. There is no attempt to incorporate "symbolic information" to communicate relevant anatomical concepts and relationships. Neither is there a specific effort given to enriching the tool with pedagogical techniques: there are no labels, drills, or quizzes to effectively present the student with clear learning objectives. Despite these limitations, it is hoped that this tool can be a useful adjunct to existing 2-D resources, and serve as a reference point for future educational endeavours in 3-D anatomy.

3.2 Previous work on ear visualization

It is important to note other previous works focusing on the visualization of ear anatomy. Non-interactive 3-D visualizations of middle-ear structures have been reported by Nomura *et al.* [Nomura *et al.* 1989]. Their work consisted of manual tracings of histological sections, surface triangulation, and Gouraud-shaded renderings. Henson *et al.* used MR microscopy to create volume rendered images of the mustached bat cochlea [Henson *et al.* 1994]. Using extraordinary device attributes (7.1 T superconducting magnets) and very long scan times (58 hours), they were able to obtain very high resolution volumes (25 μm isotropic) of this minute structure (4 mm in diameter). Figure 3-1 shows their images of the cochlea rendered at various angles.

Researchers at University Hospitals (Leuven, Belgium) have reported the use of high-resolution, spiral-CT imaging of the ossicular chain in live human patients [Hermans *et al.* 1995]. The slice thickness of their machine is normally 1mm, but by overlapping slices scanned at 0.1 mm intervals, they were able to achieve very high resolution images. In some cases, their 3-D reconstructions provided additional information not available from the standard analysis of 2-D CT images. They report, however, that the time and effort required

at the moment to segment the ossicles from each slice and reconstruct the surfaces might be warranted only for difficult diagnoses.

Another work that, like this thesis, uses the VRML standard to describe and view ear structures, has recently been carried out the Department of Otolaryngology at the University of Wisconsin. There, Joe Toejeck, a Ph.D. candidate in educational technology, has created a model of many of the middle and inner ear structures. Although the resolution (unspecified) does not portray fine detail, it is a very complete and large (> 5Mb in size) model of the outer-, middle- and inner-ear structures [Toejeck 1996].

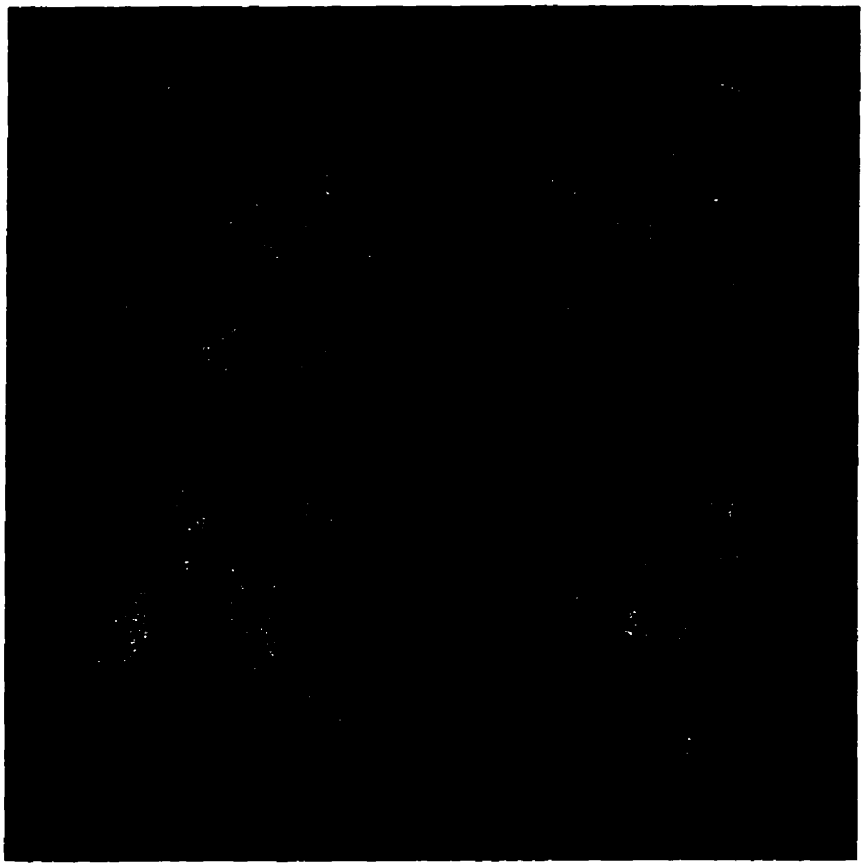


Figure 3-1: Volume rendering of the cochlea of a moustached bat.

3.3 Comparison of human and cat middle-ear anatomy

The histological data used in this project were derived from a cat cadaver. Since our work is directed at the medical school anatomy curriculum, it is worthwhile to compare human middle-ear anatomy with that of the cat. Figure 3-2 shows schematic coronal sections of cat and human middle ears. A striking feature when comparing this region in the two species is their similarity in size and structure. It is noteworthy that the sizes of middle-ear structures tend to vary less from species to species than does overall body size [Funnell *et al.* 1982]. Due to the close resemblance of these middle ear structures, data from cat specimens have often been used for modelling and experimentation [Funnell *et al.* 1978, Funnell 1983, Funnell 1987, Funnell *et al.* 1992]. The main differences in the case of the cat are the additional bulla of the middle-ear cavity, and the absence of the mastoid bone air cells.

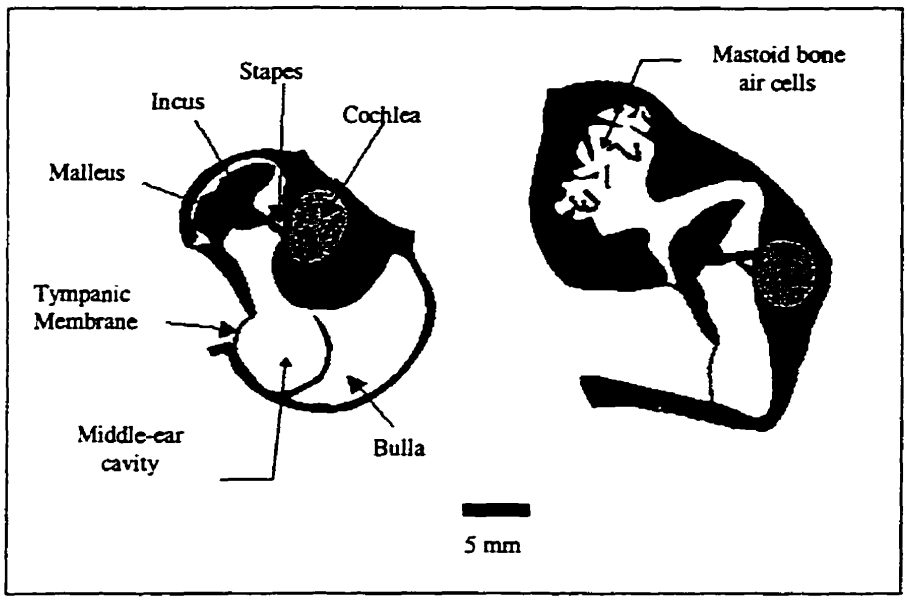


Figure 3-2: Comparison of schematic coronal sections of the middle-ear region for the cat (left) and human (right) [from Funnell 1989]

4. 3-D Reconstruction from Histological Sections

4.1 Introduction

This section describes the procedures used to acquire the histological data and process it into a form suitable for 3-D display. The steps involved are illustrated in Figure 4-1. The preparation of the slides, their images, and some of the 3-D reconstructions had been done previously. Elements of the processing added by this thesis are shown. First the cat cadaver head was embedded in plastic and sectioned using a conventional microtome apparatus. The sections were then stained and mounted on glass slides and photographed with 35 mm film. These images were subsequently scanned and stored in Photo CD format. The resulting images were aligned using a cross-correlation algorithm, and described in VRML format.

Two methods were used for surface reconstruction: the malleus and tympanic-membrane structures were created by manual contour tracings of projections of the glass mounted slides, followed by a meshing algorithm that connected the contours with triangles [Funnell *et al.* 1992]. The middle-ear cavity model was segmented by identifying regions corresponding to the cavity for each section, using a flood-fill tool. The resulting binary volume was used to create a surface using the marching-cubes algorithm. In either case, polygon-count reduction and conversion to VRML format completed the processing. VRML translation is discussed in Chapter 5.

The steps of the surface reconstruction will be described in detail below.

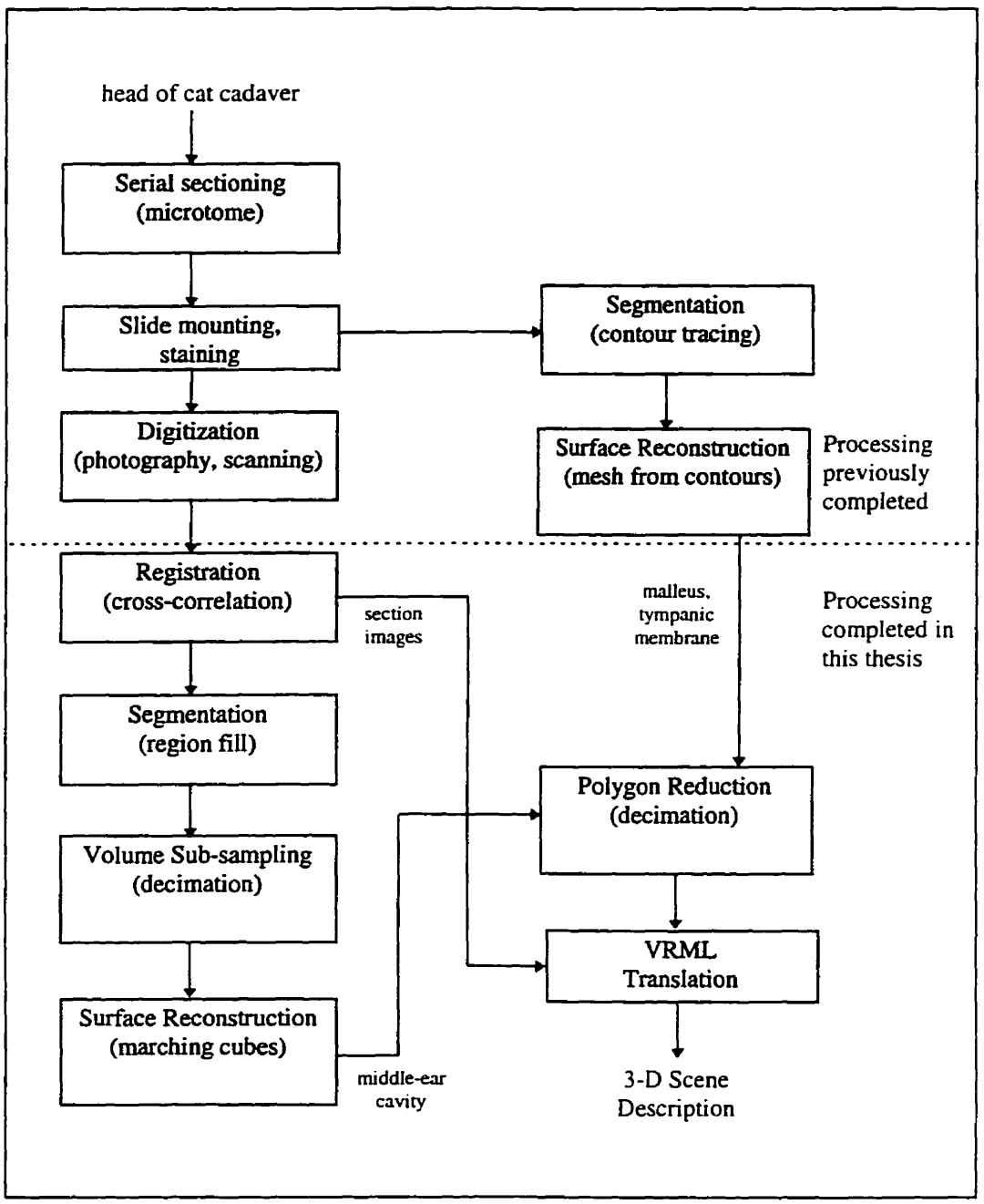


Figure 4-1: Procedures involved in acquiring the histological sections, and processing them into VRML 3-D scenes.

4.2 Histological sections

Stained histological sections have been a standard source of high-resolution anatomical images. Producing them, however, is a difficult task and has numerous drawbacks. Sections tend to be time consuming to process (it may take weeks to months to complete), and

delicate to handle (sections are frequently lost or damaged). It is also quite difficult to obtain reference landmarks to orient the slices for subsequent registration: in the absence of such contextual information, the slices must be aligned by methods that are not fully reliable (e.g., 'by-eye' manual methods, algorithmic pattern matching, etc.). It is also generally the case that the slicing thickness is significantly greater than the final resolution of the in-plane images, causing aliasing effects due to the insufficient sampling resolution of the slicing dimension. This is in contrast to MRI or CT data, where all dimensions can have the same resolution (*isotropic volumes*), and in any chosen plane all dimensions are equally accurate once appropriate transformations are performed.

The histological sections used in our work were obtained from approximately 300 slices of a cat specimen. The sectioning occurred in the transverse (axial) plane at 30μm intervals, and every other slice was mounted on slides. The slides were then photographed and digitized, producing a final resolution of 3072 × 2048 pixels, and a field of view of 45mm × 25mm. Thus the digitized set of images had a resolution of approximately 12μm × 12μm × 60μm (i.e., a 5:1 ratio of in-plane resolution to slicing thickness).

Even the reduced resolution of the slicing dimension is significantly greater than the resolutions normally possible with CT and MRI (compare to Table 1: Resolutions of the Visible Human dataset). However, using special techniques with in-vitro specimens, both of these modalities are becoming more comparable to the resolution of histological sections. We have made use of high-resolution CT data of the human middle ear from S. Gilani of Stanford University and the John P. Robarts Research Institute at the University of Western Ontario [Holdsworth *et al.* 1993], having an approximate resolution of 120μm × 120μm × 130μm. We have also experimented with very-high-resolution MRI data having resolutions on the order of 25μm × 25μm × 25μm [Henson *et al.* 1994]. This resolution may improve further, although estimates of the theoretical lower limit of MRI voxel size range from 1 to 10μm [Callaghan and Eccles 1987].

4.3 Image registration

4.3.1 Background to problem

The processes of slide preparation and digitization of the cat histological sections introduced unknown translation and rotation components into each section image. In order to align and superimpose these sections with the 3-D VRML model, it is essential to first align the sections themselves. Since there are no fiducial markers providing landmarks, it is necessary to find algorithmically the transformation which accurately estimates the best alignment between the sections.

The loss of contextual information that fiducial markers provide, however, means that any algorithmic alignment (which is generally some form of pattern-matching) can introduce shearing and torsioning effects. Some positional and rotational differences between sections are actually correct: best matching can actually 'over-align' sections for structures that are not axially symmetric [Schwartz *et al.* 1988].

Fortunately, the cat head slices do exhibit some axial symmetry (about the centre line): by examining the alignment in both ear cavities, for example, it was hypothesized that a compensating effect from the contributions of each side would occur, reducing error.

Before describing the approach used in our work, a brief description of some of the possible alignment methods will be given.

4.3.2 Standard approaches

One of the simplest alignment methods is principle-axes transformation. The alignment procedure for two images superimposes the centre of mass (centroid) and the principal axes for each image [Hibbard *et al.* 1992]. Deubler *et al.* report a wavelet-based approach for alignment of serial sections [Deubler *et al.* 1995]. This method also considers global and local deformations in the transformation estimate and exploits the multi-resolution nature of wavelet analysis to significantly improvement the performance of the algorithm.

Pattern matching analysis has often used cross-correlation to detect the presence of a template image within a larger image. The cross-correlation of the two images is used to form an estimate of the translation or rotation shifts required to align the images [Pratt 1978, Hibbard and Hawkins 1988, Hibbard *et al.* 1992]. This method is the most popular of the alignment approaches because it tends to be more robust than the principle-axes method, and lends itself to a simpler implementation than the wavelet approach. In the case of our sections, the geometric distortions of the section images were assumed to be negligible, so that the wavelet method was considered an unnecessarily complex approach. For these reasons, the cross-correlation method was chosen for our work.

4.3.3 Theory of cross-correlation registration

There are many possible methods for measuring the degree of similarity between two functions f and g over a region S. Two examples would be:

$$\iint_S |f-g| \quad \text{or} \quad \iint_S (f-g)^2 \tag{4-1}$$

Using the second expression, the mean-square difference (or energy difference), it is possible to develop further insight into measuring the function match. If this expression is expanded, we have

$$\iint (f-g)^2 = \iint f^2 + \iint g^2 - 2\iint fg \tag{4-2}$$

Thus, if we are given a template function f, and wish to find the window region in g that maximizes the match between f and g (i.e., we wish to minimize eqn.(4-2)), $\iint fg$ is a measure of that match.

However, this measure is sensitive to the template and window energies. If the template has regions of low energy, for example, matching regions of low energy in the window will result in a low measurement. Now, from the Cauchy-Schwartz inequality, we have

$$\left|\iint fg\right| \le \sqrt{\iint f^2 \iint g^2} \tag{4-3}$$

with equality occurring if and only if $g=cf$. If we then normalize this expression as

$$\frac{\left|\iint fg\right|}{\sqrt{\iint f^2 \iint g^2}} \le 1 \tag{4-4}$$

we obtain a metric that is insensitive to both the template and the local window energy.

Applying these ideas to the discrete case of matching a reference image g and template image f displaced by (m,n), the measure of matching is given by

$$\rho(m,n) = \frac{\sum_j \sum_k f(j,k)g(j-m,k-n)}{\sqrt{\left[\sum_j \sum_k |f(j,k)|^2\right]\left[\sum_j \sum_k |g(j,k)|^2\right]}} \tag{4-5}$$

where ρ has a range of $-1 \le \rho \le 1$ and a value of $\rho=1$ occurs if and only if the images are identical. The summations are calculated over the region of overlap, as shown in Figure 4-2. The expression in the numerator can be recognized as the cross-correlation $R(m,n)$ of images f and g, and for this reason, ρ is referred to as the *normalized cross-correlation* (also known as the correlation coefficient) [Pratt 1978, Rosenfeld and Kak 1981, Hibbard and Hawkins 1988, Jain *et al.* 1989].

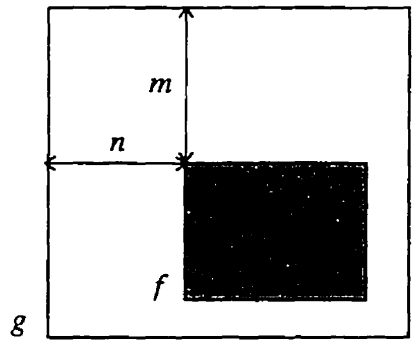


Figure 4-2: Matching of template image f and reference image g. Images are compared in the shaded region of overlap.

By dividing by both the local window energy and the template energy, a normalized value is produced that can be used to compare relative matching success. In practice, the cross-correlation is usually calculated more efficiently by Fourier-domain filtering [Hibbard and Hawkins 1988]:

$$R(m,n) = F^{-1}\left\{F(f) \cdot F(g^*)\right\} \tag{4-6}$$

The problem of aligning two images with unknown translational differences can be solved by forming the cross-correlation between the image pair, searching for the cross-correlation peak, and translating one image with respect to the other based on the peak value.

The window energy of the reference image varies slowly with (m,n) and it is normally unnecessary to explicitly calculate it, especially if the translational offsets are typically small compared with the template size. Instead, the mean grey level of each image is subtracted from each pixel value before computing the correlation. This reduces the sensitivity to grey-level scaling of the image data and sharpens the maximum peak [Gonzalez and Wintz, 1977].

It is a straightforward extension of this development to determine an estimate of the rotational difference between two images. Since rotation in rectangular Cartesian coordinates is equivalent to translation in polar coordinates [Pratt, 1978], polar-resampling

of the images allows them to be aligned by the same algorithm used for the translational alignment [Hibbard and Hawkins 1988, Hibbard *et al.* 1992].

When performing the cross-correlation for translation by Fourier filtering, it is necessary to augment the image sizes to avoid convolution wraparound error. For two images of size $M_1 \times N_1$ and $M_2 \times N_2$, each image must be increased to $(M_1+M_2-1) \times (N_1+N_2-1)$. However, for rotational alignment, it is possible to take advantage of convolution wraparound (due to the periodicity of the polar-resampled image) and eliminate the need to augment image sizes. This reduces computational load significantly, since the FFT routines represent the bulk of the processing time. In our case, $M_1 = M_2$ and $N_1 = N_2$, so avoiding a two-fold augmentation in each dimension avoids a four-fold increase in processing time, since standard 2-D FFT algorithms such as vector-radix have a computational complexity of $O(n^2)$, where n is the image size in one of the dimensions [Dudgeon and Mersereau 1984].

Translational and rotational alignment are coupled in such a way that neither can be performed independently. For a pair of slightly dissimilar images (such as serial sections), an iterative procedure of alternating translational and rotational alignments will often converge to a point which maximizes the correlation coefficient. Failures can occur when, for example, the template size is too small (insufficient comparison area), or the required translation has a magnitude comparable to the template size. Thus it is advantageous to use a template size that is as large as possible.

4.3.4 Methods

The rotational and translational alignment procedures were implemented as Matlab routines. The large memory requirements required by cross-correlation and the inefficient memory management of the Matlab package [Mathworks 1992] soon overwhelmed even the virtual memory resources of our DEC Alpha UNIX system (64Mbytes RAM and approximately 500 Mbytes virtual memory), so that it was not possible to align at the full resolution of the images (3072×2048). At this resolution, storage of each image alone would require

approximately 50Mbytes using eight-byte floating-point arithmetic. For this reason, a four-fold reduction in resolution was used (768×512), with an even smaller template window size (600×200). The resolution of our data set was therefore reduced from 12μm × 12μm × 60μm to 48μm × 48μm × 60μm. The template dimensions were chosen to be as small as possible for computational purposes, but large enough to ensure coverage of the regions of interest (the middle-ear cavities), where axial symmetry occurs.

The polar-resampling was performed using a bilinear interpolation [Castleman, 1978]. Experimentation indicated that achieving angular resolutions of 0.5° (720 points) required only coarse radial sampling (typically 16 points). Finer radial sampling (32 or 64 points) prolonged processing (again, due to the $O(n^2)$ FFT complexity), and did not significantly increase precision in the alignment estimate. At this sampling density, the resampling time was insignificant compared with the times required for forward and reverse Fourier transforms.

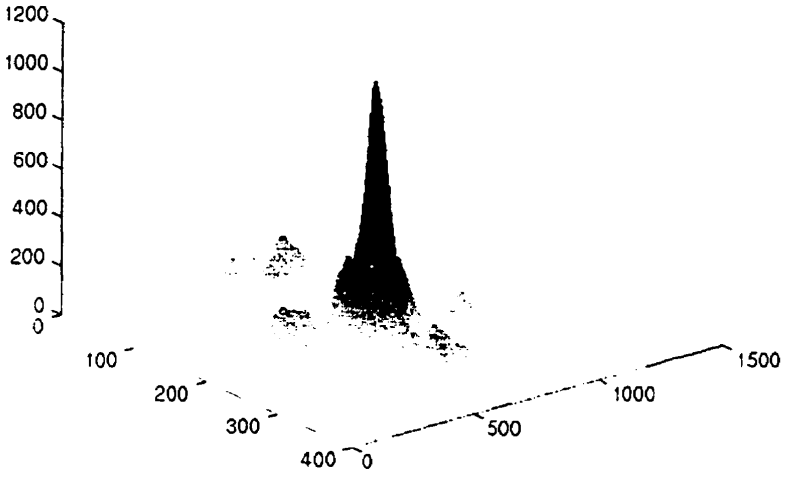
The rotation estimate is dependent on the choice of origin for the polar sampling. Therefore, if there is some translational misalignment, the image centres will not coincide, and an inaccurate angular correction can occur. Hibbard reports more robust rotation estimates by first aligning the autocorrelation of each image as a rough estimate, before using the images themselves for subsequent estimates [Hibbard and Hawkins 1988]. The shift-invariance of the autocorrelation function removes the effects of any translational misalignment, but its smoothing effect makes it inappropriate for fine alignment. Using this approach with our data, was in fact found to be costly and unnecessary for convergence. The initial translation shift tended to reduce the translational misalignment enough to minimize the impact on the subsequent rotational estimates.

The peak of the cross-correlation at each stage was used to choose the alignment estimate. The normalized correlation coefficient was then calculated for this peak value only since, generally, calculating the window energy for every possible translation is prohibitively expensive. The iterative process was then continued until the translational and rotational

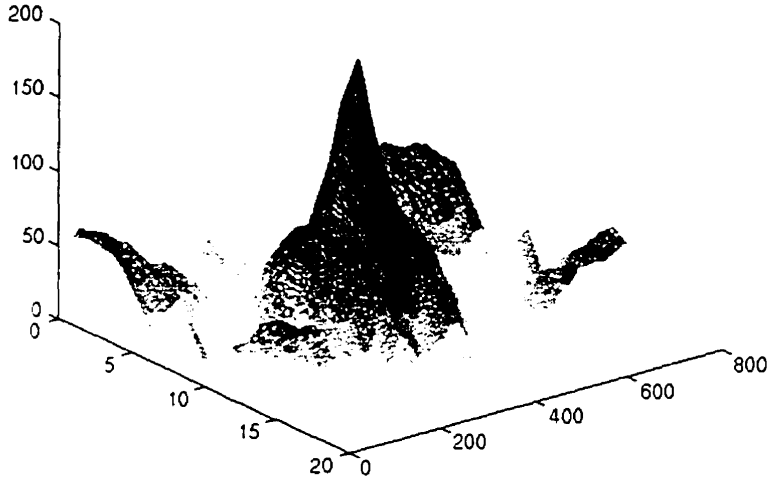
shifts were less than the spatial resolution (1 pixel for translation and 0.5° for rotation). The aligned image at this stage was then used as the reference image for the next section alignment.

4.3.5 Results

The peaks of the cross-correlation function for typical translation and rotation estimates are shown in Figure 4-3. In Figure 4-4, two sections requiring alignment are shown, as well as the original location of the two regions of comparison. The results of the alignment for this pair are also shown. The reference image is in red and the matched image is in green. The symmetry of the shifts is especially apparent in the middle-ear cavities where symmetrical displacements of the cavity wall appear as green and red fringes. Processing of each pair of images required approximately 10 minutes of CPU time on our hardware.



(a)



(b)

Figure 4-3: Typical peaks of the (a) translational and (b) rotational cross-correlation functions.

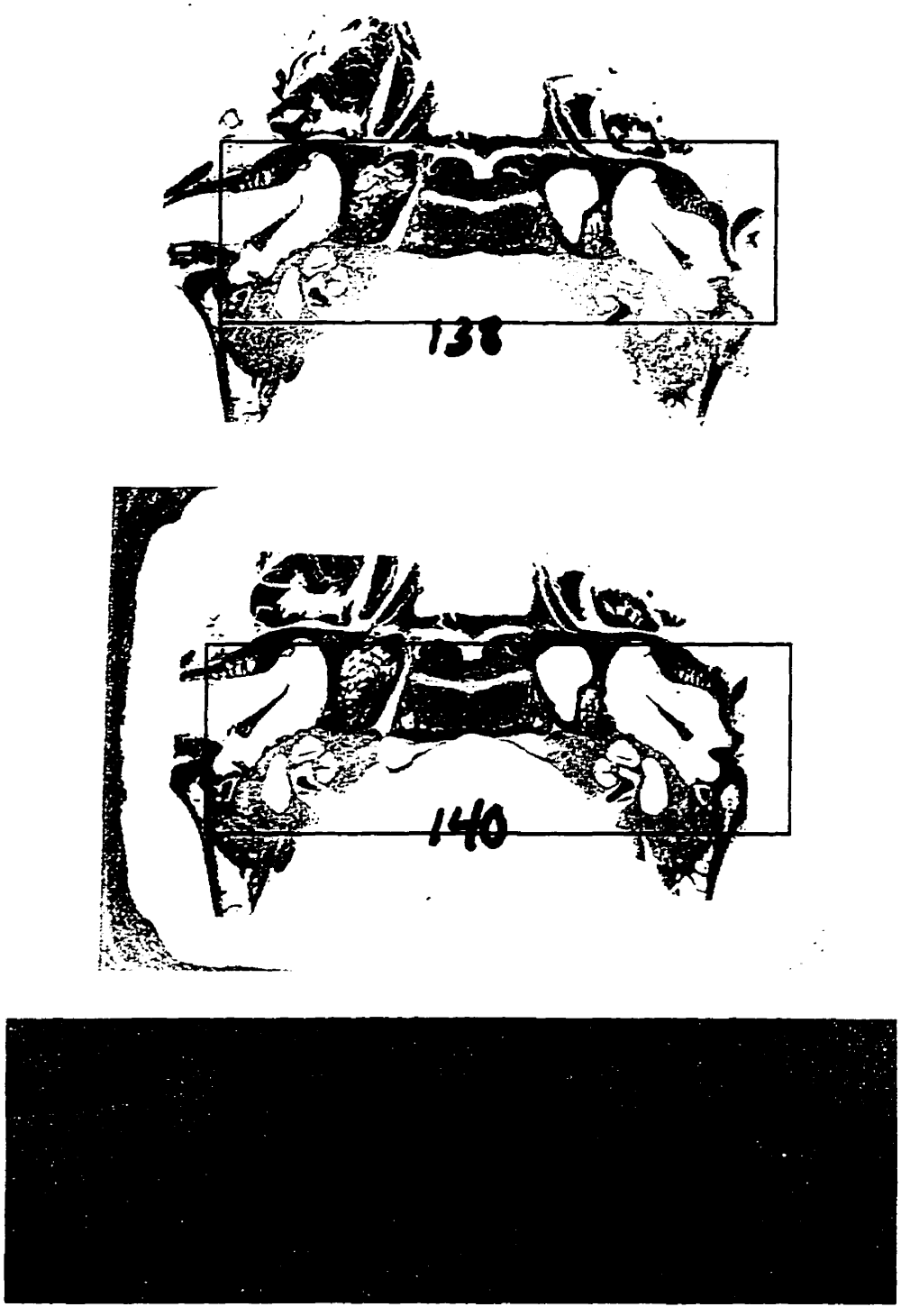
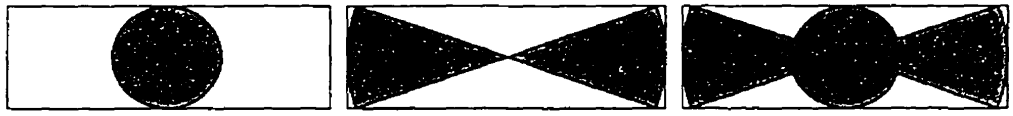


Figure 4-4: Two consecutive histological sections (superior view) and a superposition of their aligned images. The first image appears predominantly in red while the second image appears predominantly in green.

Success at each stage was evaluated by subjective observation of the overlaid images: symmetrical shifts and distance uniformity between associated contours were considered to be good indicators of a reliable registration. The correlation coefficient, a measure of the degree of overall matching between images, also gave some indication of success: before alignment, typical values were 20-30%. The final correlation coefficient after a successful alignment was typically in the range of 70-90% and significantly less (30-50%) for false matches.

While the translational alignment can use the full 600×200 template image for comparison, the polar resampling required by rotational alignment is more conveniently implemented by sampling radially symmetric geometries rather than the entire rectangular area. The simplest such geometry is a circular area with a diameter equal to the length of the shortest image dimension (Figure 4-5(a)). Several cases failed on the rotational estimate with this approach, however, due to the small image area used for comparison and the relatively large initial translations required (compared with the template size). In these cases, if a template area was manually selected that approximated the first required translation, the processing could continue successfully. However, the frequency of this problem (approximately 10 false matches of this type occurred), prompted an investigation of the use of other geometries that possess greater bias towards the horizontal extremes, where the middle-ear cavities are found, and include larger area coverage. In fact, much more robust results were observed when two symmetrical sectors (Figure 4-5(b)) were used: convergence was completely successful for all of the problematic cases, and no further manual intervention was required. Figure 4-5(c) suggests another possible geometry that combines the first two areas, presumably yielding an even greater reliability. This case (essentially four sectors) would require a radial sampling strategy that ensures rectangular arrays: zero padding outside the sectors of shorter radii would be the simplest example.



(a) (b) (c)

Figure 4-5: Three possible area selections for rotational alignment. Case (b), with greater total coverage area, converged more successfully than (a). Case (c), not used, combines (a) and (b) for even greater coverage and, presumably, rotation estimates with even greater reliability.

To give some indication of the gross alignment of all the sections, Figure 4-6 and Figure 4-7 show a series of aligned images, one spanning the entire 130 images at 16 slice intervals, and the other showing 16 consecutive images. Figure 4-8 shows a cross section of data in three orthogonal planes (cropped to include only one middle-ear cavity), giving some indication of the local alignment success and the overall trend of the alignments. Contours in the two planes orthogonal to the slicing plane (the top images of Figure 4-8) are clearly apparent and appear smooth (apart from noise on the order of a few pixels). Non-uniformities appearing as horizontal bands are attributable to variations in the staining intensity, as it is difficult to maintain constant stain concentration for every section.

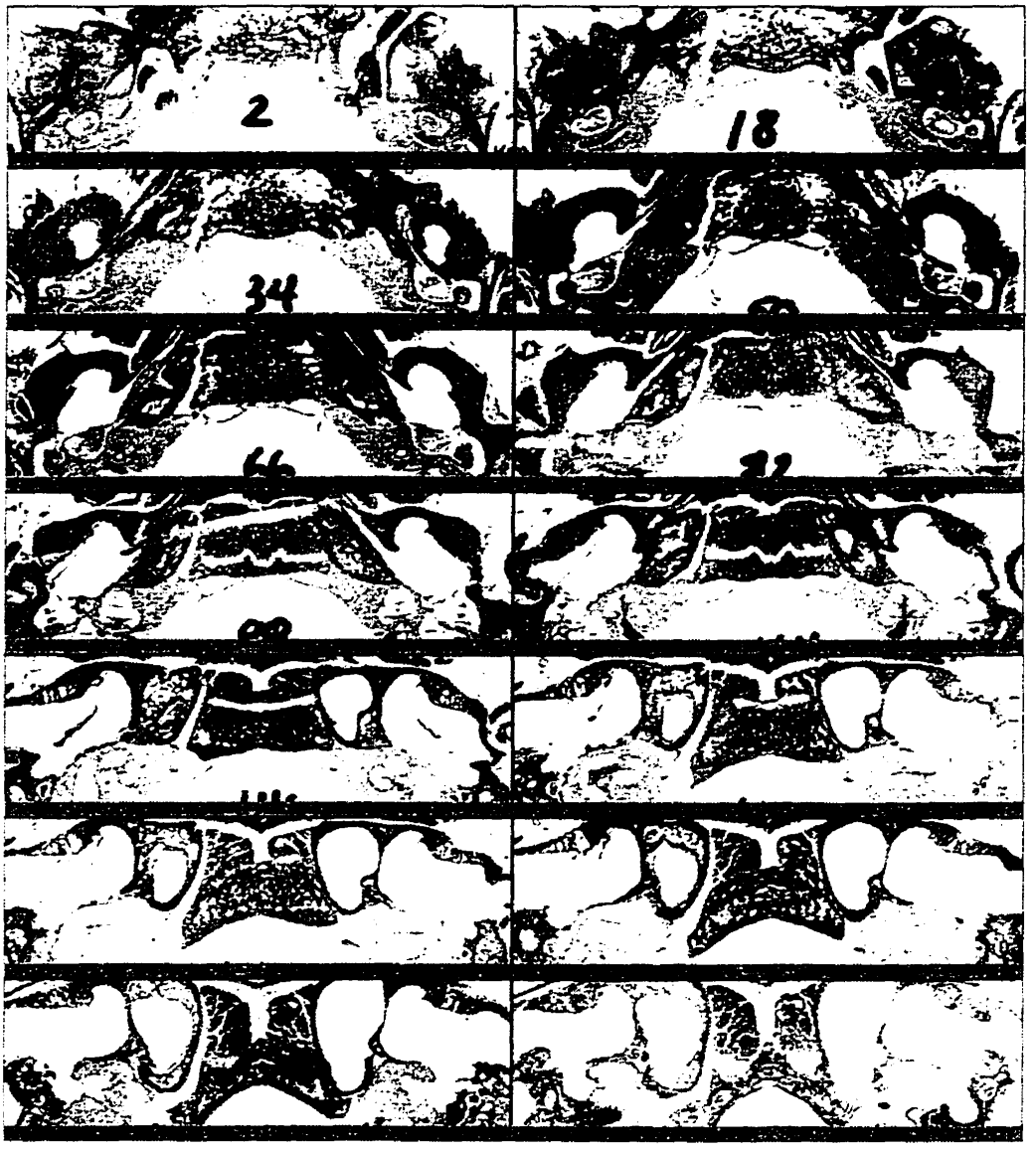


Figure 4-6: Series of aligned images, spanning entire dataset (each pair 16 slices apart)

Figure 4-7: Series of consecutive aligned images.

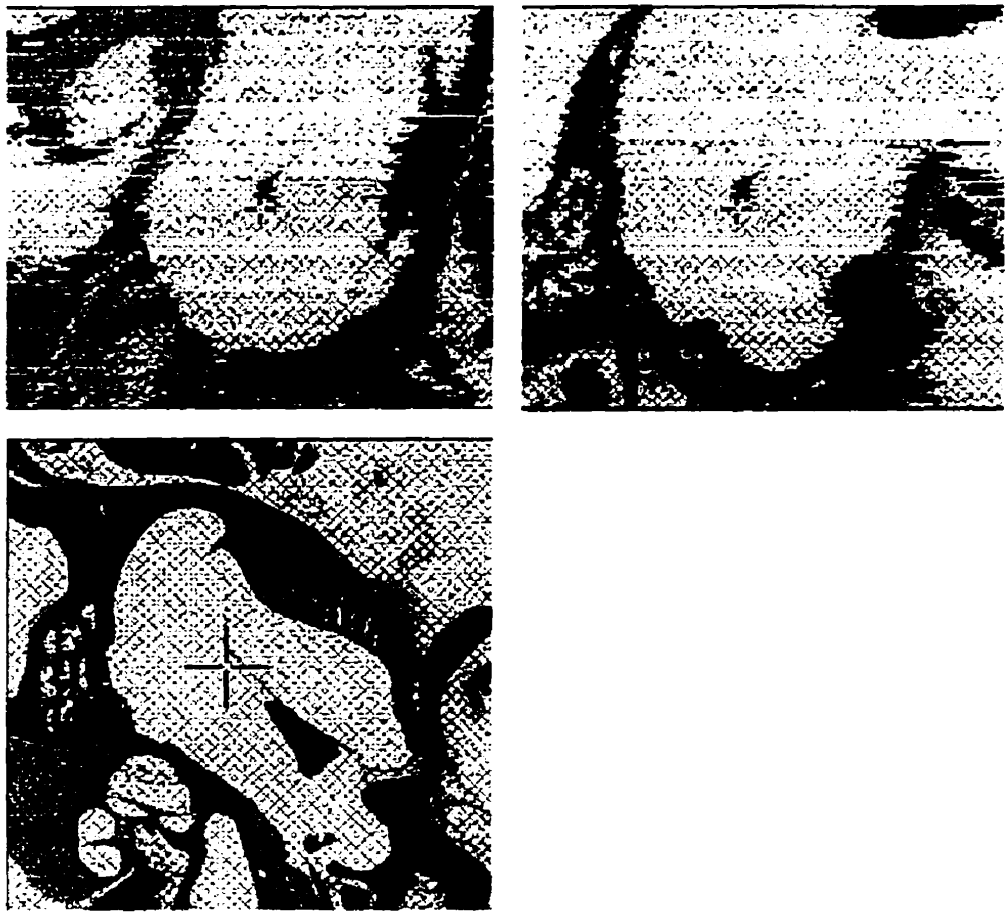
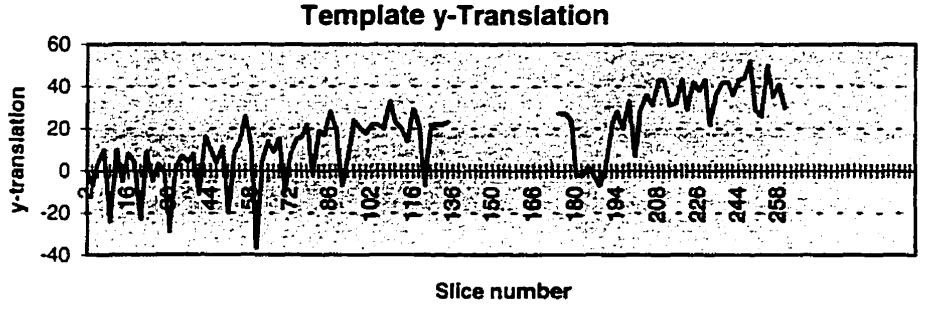


Figure 4-8: Cross sections in three orthogonal planes of the registered data. The cross in each image indicates the location of adjacent slicing planes.

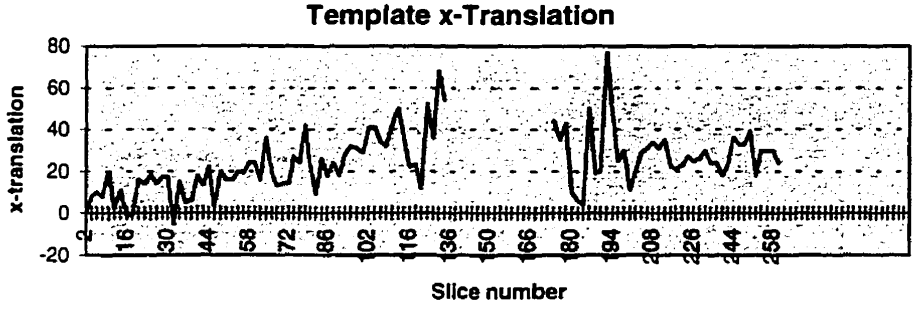
Figure 4-9(a) and (b) show the x and y translations required to align the sections at each stage. The total translation required for each alignment was typically less than 20 pixels in either dimension. While data from slices 132-174 were not available for these plots, it seems apparent that a gradual drift upwards is occurring in both the x and y translations. Over the 130 slices, an estimate on this trend from the plot would be about 30×40 pixels. This may be an indication that the registration has introduced shearing on the order of 15-20% in each of the two dimensions, or about 1.44-1.92mm over the 7.7mm range of the slices. It is also possible that this shearing is correct, and was present in the original data.

The inability to ascertain which is true is a drawback of any registration attempt that does not include reference markers.

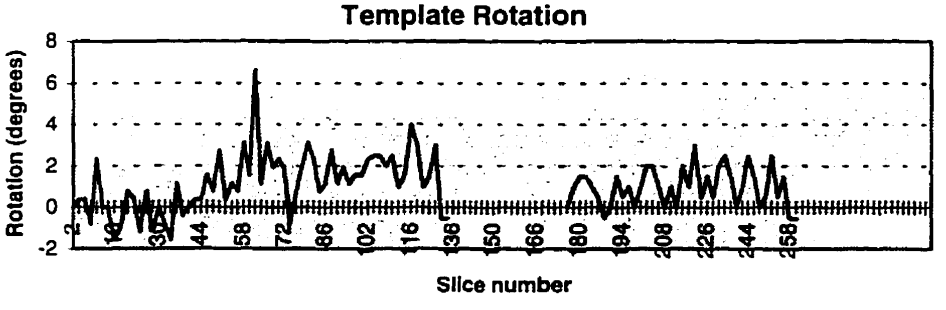
The total rotation required at each alignment was typically less than 3 degrees. Figure 4-9(c) shows the total slice rotation at each alignment. In this case, a slight trend of angular accumulation is apparent, on the order of 4 degrees over the set of sections (note that the accumulation was reset to zero after the missing sections). This may be an indication of the degree of axial torsion created by the alignment procedure, although again there is the possibility that this torsion was present in the original data. To further examine the global effects, two images, 64 slices apart, were superimposed and the angles between centre-lines compared (Figure 4-10). Assuming the centre-lines are of constant orientation, this angle of 3° gives some estimate on the bounds of rotational accuracy of the technique. Due to the lack of centre-lines in some of the beginning sections (2-100), it was not possible to compare a wider separation of slices. However, if the value for 64 slices is extrapolated to the full 130 images, an estimate on the error bound would be approximately 6° degrees from one end of the sections to the other, roughly consistent with the 4° estimate from the trend-line. This gives us some confidence in our estimate of the error bounds of the registration technique.



(a)



(b)



(c)

Figure 4-9: Overall alignment results for the serial sections. (a) and (b) show the x and y translations required to align the serial sections. (c) show the corresponding rotations for alignment. Note that slices 132-174 were not included for each of these graphs.

Figure 4-10: Overlay of aligned images for two sections 64 slices apart. The overall rotational error of 3 degrees was estimated by measuring the angle between respective centre-lines, which are assumed to have constant orientation throughout the slices.

4.4 Surface reconstruction

4.4.1 Surfaces from contour tracing

Two approaches were taken to create surface reconstructions from the histological data. Work previously done had formed surfaces from manually traced contours. The contours were traced by projecting each of the slides onto a digitizing tablet. These contours were then manually aligned and processed by a meshing algorithm. The mesher sampled each of the contours at prescribed intervals and created surface triangulations between sampled points of successive contours, using a cost minimization procedure [Funnell 1984]. From this process, a full model of the malleus and a partial model of the tympanic membrane were created [Funnell *et al.* 1992]. The tympanic-membrane model was incomplete because the prepared section data did not span the full extent of this structure.

A partial model of the middle-ear cavity was also created by this same process. This model was also incomplete due to the limited extent of the data. It was also of unwieldy size because of the large number of polygons required to describe it. Its surface area is much larger than that of the malleus, so in order to have a surface description that does not overwhelm memory during processing, and has reasonable rendering times, some data reduction was required.

4.4.2 Surfaces from marching cubes

It would have been possible to reduce the data for the cavity surface by modifying the processing units described above: the sampling intervals for contour points and/or slice planes could have been increased, producing a smaller reconstruction with some loss of detail. Instead, however, a different approach was taken that was more volume-oriented, and that used publicly available software tools.

4.4.2.1 Theory

The marching-cubes algorithm [Lorenson and Cline 1987] has become a widely used method of extracting surfaces from volume data. The user first specifies a contour level defining the desired surface, which can be a known parameter of the modality (the absorption value corresponding to bone in CT, for example) or a range of values spanning a surface transition. In the case of the histological sections, if a structural region can be identified across a series of sections, a binary segmentation can be achieved (i.e., every image pixel is classified with respect to its region membership), providing a simple threshold for surface definition.

The algorithm commences by moving ('marching') to one voxel ('cube') at a time and performing a binary classification of each *vertex* of the current voxel. By classifying each vertex, the algorithm is able to generate a finer and more accurate surface compared with schemes based on edge detection or global thresholding (based on *spatial-occupancy enumeration* volumes that resemble a construction of children's building blocks), since the surface intersection points are interpolated on the lattice, rather than being restricted to the lattice points themselves. Since each voxel has eight vertices, there are 2^8=256 possible ways in which the surface can intersect the voxel, of which 15 are unique (see Figure 4-11). Usually a look-up table of these cases is used to simplify the surface identification of each transition voxel. Each of these cases corresponds to a small number of polygons that form the surface definition. One advantage of this technique (which was not exploited in our work) is that it easily incorporates an accurate normal estimate by calculating the local

gradient at each vertex. This can later be incorporated into the shading calculations for a more accurate and smoother surface display.

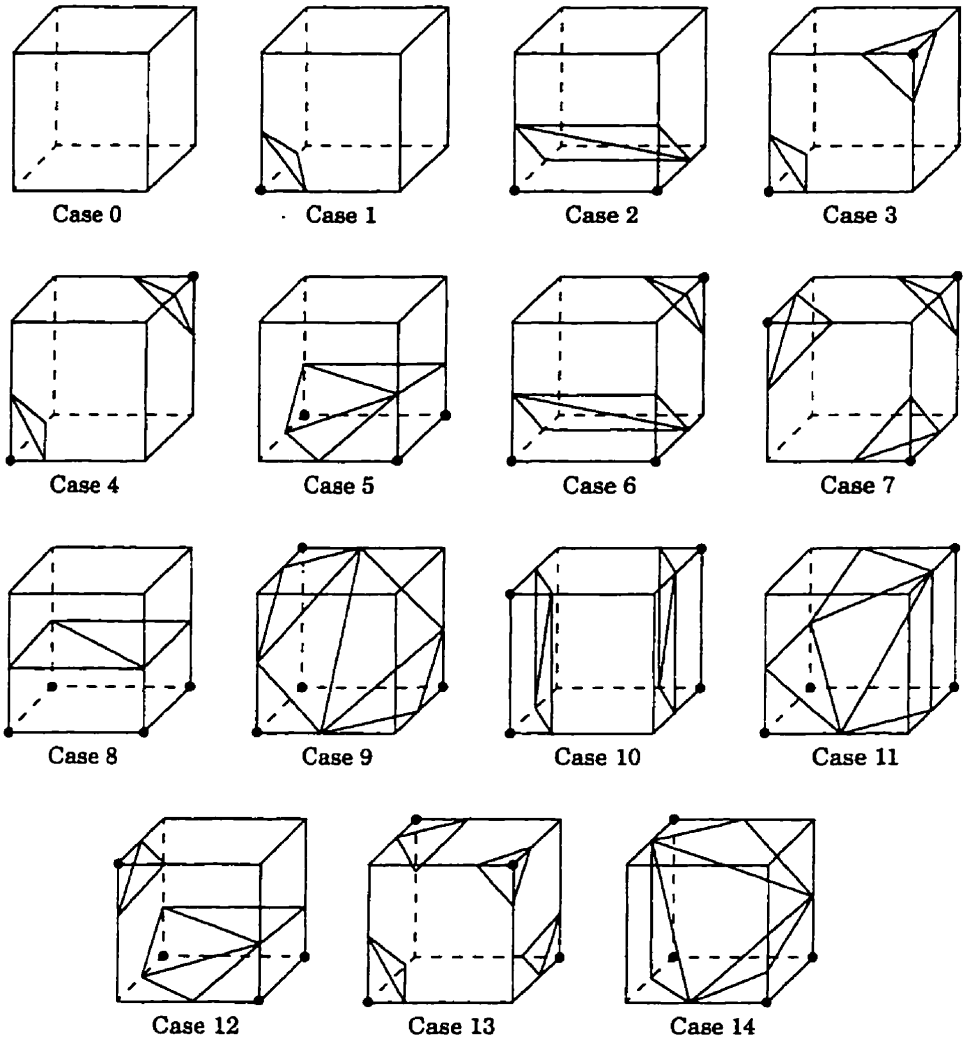


Figure 4-11: Marching cubes surface intersection cases (from [Schroeder *et al.* 1996]).

While marching cubes can produce very accurate surface representations, it does so at the cost of high data bandwidth: since the mesh polygons it creates are of comparable size to the voxels, regions of low surface curvature may be described by many redundant polygons. It is not uncommon for typical volumes (128^3 voxels, for example) to produce hundreds of thousands of surface polygons. In order to achieve polygon models that are of reasonable

size for rendering environments, it is often necessary to reduce the volume resolution before processing with marching cubes [He *et al.* 1995].

Sub-sampling of this sort will reduce the number of polygons uniformly across the surface. Mesh processing that can adapt to regions of low surface curvature can further reduce the polygon count. One strategy is *polygon decimation* [Schroeder *et al.* 1992]. This topology-preserving procedure (i.e., it preserves the complexities of holes and *non-manifold* topologies: see Figure 4-12) reduces the total number of polygons in a mesh, and forms a good approximation to the original geometry. The choice of which points to delete is determined by a decimation criterion, a measure of the local error introduced by the point deletion. This measure is the distance of the point to be deleted from a local plane that has been approximated from its connected vertices. Once an identified point is deleted, the resulting hole defined by neighbouring vertices is triangulated. To preserve edges, they are identified and are subject to the decimation criterion during triangulation: they cannot be moved beyond this distance. Schroeder reports reduction factors of up to 90% for certain datasets. An iterative reduction process can continue until some threshold distortion level has been reached. Renze and Oliver have recently generalized this approach for the reduction of volume tetrahedizations [Renze and Oliver 1996]. In both the 2-D and 3-D cases, they use a form of constrained Delauney tesselation to re-mesh the holes introduced by point deletions.

(a) (b)

Figure 4-12: Examples of (a) locally manifold and (b) locally non-manifold topologies. Topology can be described by a list of atomic vertex sets called *cells* (polygons, triangles, etc.). In 2-D, if every edge of a 2-D cell is used by exactly one other cell, than the surface is locally manifold. In 3-D, if the face of a 3-D cell is used by exactly one other cell, then the region is locally manifold [from Schroeder *et al.* 1996].

One weakness of polygon decimation is that the error metric measures the distance between a deleted point and an intermediate approximation of the mesh, and not the original mesh. There is therefore no certainty as to the accuracy of the reduced mesh. This is of concern to physicians involved in certain areas of patient therapy (surgical planning, radiotherapy, etc.) that demand fidelity in the presentation of anatomy. Unlike polygon decimation, the *superfaces* algorithm is a one-pass approach that provides precise error bounds on the final mesh [Kalvin and Taylor 1996]. Starting with a randomly selected face, faces that satisfy an error criterion are merged into surface patches (*superfaces*): the merge occurs if, when constrained to movement below the error criterion, the vertices of the superface and a trial face can be fitted onto a plane. If there is no solution to the associated set of equations, the superface is complete. Because an infinite number of planes can potentially satisfy these conditions, more merging may occur than is possible with another related strategy, *geometric optimization* [Hinker and Hansen 1993], which controls merging with a single approximating plane, similar to polygon decimation. Once one superface has been completed, another (unmerged) face seed is chosen, and the merging process continues until all faces are considered. The edges of the superfaces are then approximated (*border straightening*) so that the edge resolution is consistent with that of the superfaces. The last step is to perform a 3-D triangulation of the superfaces. This algorithm has an efficiency comparable to polygon decimation, since the merging process, which dominates processing, can be done in $O(n)$ time. The reduction factor can equal or surpass that of polygon

decimation, although the incompatibility of the error metrics used in each case make direct comparisons difficult.

4.4.2.2 Methods

While marching cubes could be directly used on the volume defined by the registered section images, it is difficult to select a threshold that successfully extracts the surface of interest. One problem is that one global threshold may be insufficient: an adaptable threshold may be required in different regions of the volume. As well, numerous distinct structures will often be extracted, although post-processing can be used to remove the extraneous surfaces if the structure of interest is obviously the largest in the volume of interest. If a surface at a specified contour level is interior to another surrounding surface at the same level, it will be obscured during display unless transparent viewing is possible. These are some of the reasons that segmentation is such an important and complex problem. One simplistic approach is to manually segment the regions before marching-cubes processing. The accuracy of this brute-force method suffers, however, from the subjective decisions required by the interactive user. Specifically, identification of the transition regions is long and tedious work that requires constant judgments of approximation. Nevertheless, in the absence of reliable automated methods, it is an approach that is widely used and often gives adequate initial results.

Starting with the registered section images (with their 48μm × 48μm × 60μm resolution), segmentation of the middle-ear cavity was completed by importing a cropped portion of the images (of size 199×200 pixels) into the tool *display*. This tool, created by David Macdonald, a researcher in the Brain Imaging Centre at McGill, allows region filling of pixels that fall within a certain range. This was done for 130 of the cropped images, creating a binary-thresholded volume.

The segmented volume was decimated using a volume scaling tool available in *volpack*, a public-domain volume-rendering package [Lacroute 1994]. This tool first low-pass filters the data with a 3-D gaussian function, allowing a reduced number of samples to be extracted with minimal aliasing.

The marching-cubes and polygon-decimation algorithms have been made publicly available in *The Visualization Toolkit* (vtk) [Schroeder *et al.* 1996]. This is a general-purpose visualization tool providing numerous data structures and algorithms to process data for meaningful display. We used this package, under the DEC Alpha UNIX environment described earlier, to process the sub-sampled volume. One other useful feature *vtk* provides is the elimination of duplicate vertices in a polygon model. This feature was important for post-processing of the marching-cubes output.

4.4.2.3 Results

Figure 4-13 shows some results of the segmentation of the middle-ear cavity. Segmenting the entire set of 130 images required approximately 4 hours of manual region filling using *display*. As expected, marching-cubes created an unwieldy number of polygons (0.5×10^6) when processing the resulting binary volume (199×200×130). Volume sub-sampling by a factor of four in each dimension reduced the volume size to 49×50×32. Some loss of detail was introduced by the smoothing effect of the volume decimation, but at the same time the process had the positive effect of reducing the abruptness of the high-frequency inter-slice transitions and averaging out some of the local alignment errors. It should be noted that performing the segmentation at the higher resolution rather than the reduced resolution increased the visual cues available for the human segmenter, and eased the process of region identification.

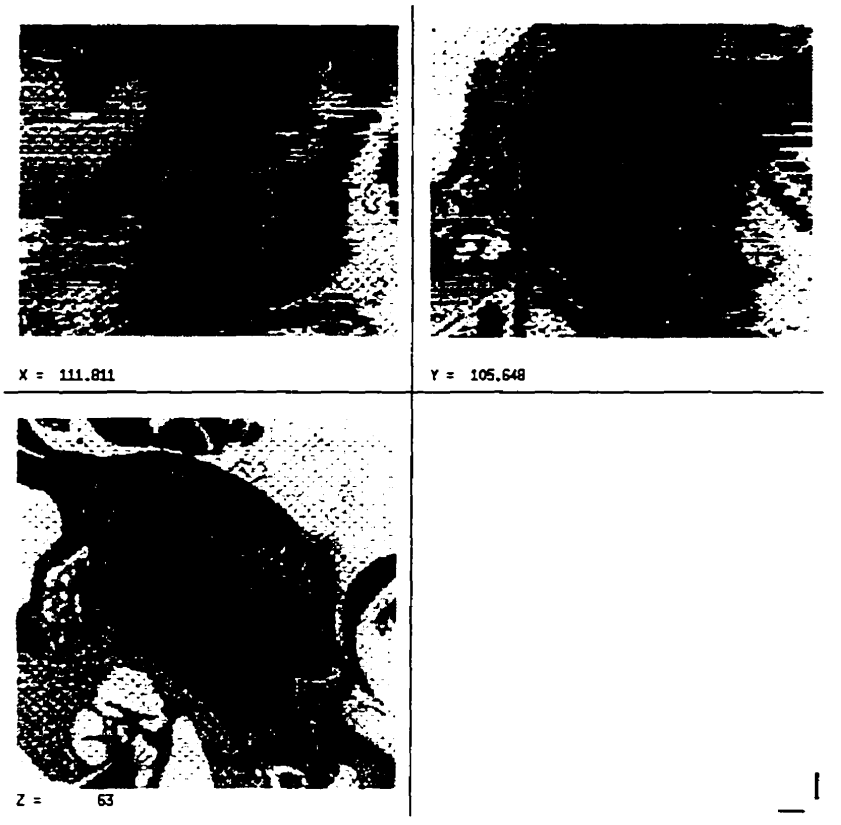


Figure 4-13: Results of segmentation. The middle-ear cavity was identified and labeled for each section (one shown at lower left), creating a segmented volume. The segmentation (regions of uniform dark grey) is also shown in the two other orthogonal planes (top).

The resulting 49×50×32 volume no longer possessed binary voxel intensities, but ranged between 0 and 1. It was therefore necessary to select a threshold that best corresponded to the original contour of the middle-ear cavity. To determine an adequate threshold value, iso-contours were identified on a representative section of the filtered volume. These contours were produced using the 2-D counterpart of the marching cubes algorithm, referred to as *marching squares*. After converting voxel intensities to byte values (i.e., range 0-255), contours corresponding to levels 1, 128 and 255 were created and superimposed on a section having the higher pre-filtered resolution. The results are shown in Figure 4-14. The support of the gaussian filter (6 voxels in each of the three dimensions) had the effect of stretching of the transition region. Level 1 overestimates the cavity volume, while at level 255 the volume is not only underestimated but has also become disjoint due to non-

uniformities introduced by the filter. The midpoint intensity at level 128 is equally weighted by points inside and outside the cavity contour, roughly speaking, and was consequently selected for the marching-cubes threshold. Due to the resolution reduction and volume averaging from neighbouring slices, however, this contour only approximates the original cavity contour. It is to be expected, therefore, that the marching-cubes surface can have only approximate correspondence to the set of cavity contours in all the slices (this correspondence will be reported in the next chapter in section 5.5.2).

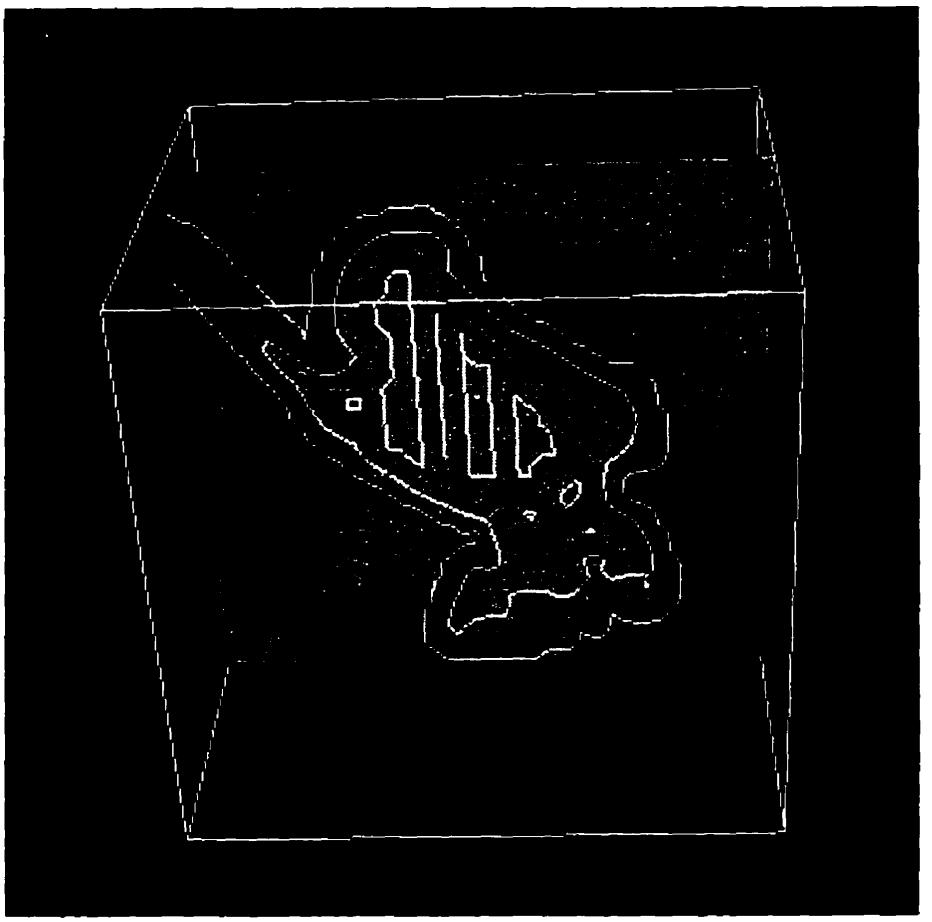


Figure 4-14: Superposition of iso-contours from the decimated volume onto the corresponding pre-filtered section image. The outer contour is intensity level 1; the middle contour is intensity level 128; the set of interior contour segments corresponds to level 255. The bounding volume of all the slices is shown in outline.

Using the threshold selected above, marching cubes created approximately 15×10^3 polygons. The polygon decimation algorithm was then used to reduce the number of polygons from 14886 to 5472, a reduction of 63%. A comparison of the meshes before and after decimation is shown in Figure 4-15. The decimation algorithm was also used to reduce the polygon count for the contour-based 3-D models. The results for all the models are shown in Table 4 below. It should be noted that the tympanic membrane, with proportionally the greatest amount of near-planar surface, achieved the greatest percentage reduction in polygon count. The resulting triangle counts represents significant improvements in the size of the polygon database to be rendered.

One problem that is apparent upon close examination of the middle-ear cavity mesh is the presence of an artifact: it corresponds to a portion of the malleus which has not been sufficiently segmented from the cavity. This problem can be addressed by refining the middle-ear cavity segmentation and re-processing the results.

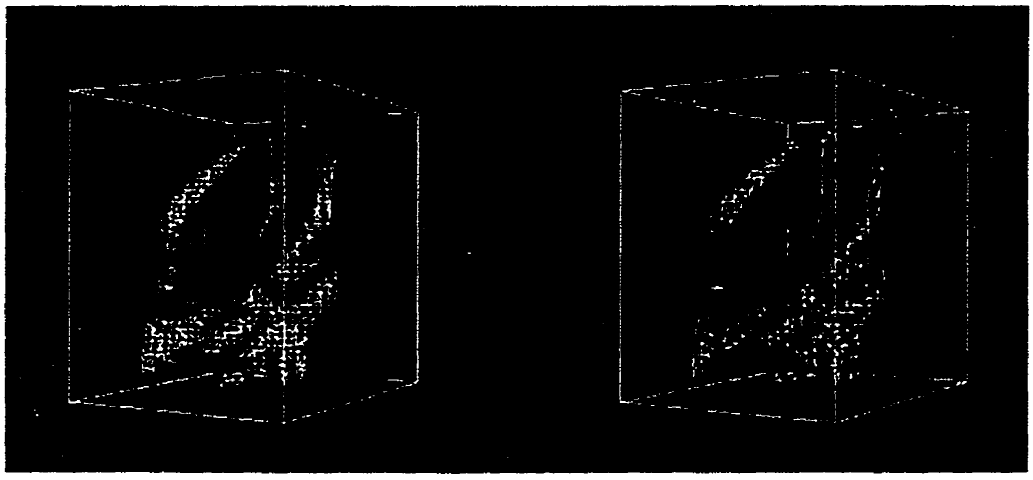


Figure 4-15: The middle-ear cavity model before (left) and after (right) polygon decimation. Interior and exterior surfaces appear as black and white, respectively. The oval opening is the ear canal, while the cut portion at right reveals the extent of the histological data. The bounding volume appears in outline.

Structure	Data Source	Original number of triangles	Number of triangles after decimation	Percentage Reduction
Malleus	Contour tracing, meshing	1260	599	52%
Tympanic membrane	Contour tracing, meshing	1159	246	79%
Middle-ear cavity	Region filling, marching cubes	14886	5472	63%

Table 4: Results of polygon reduction using the polygon decimation algorithm.

4.4.3 Discussion

The method described for surface reconstruction from marching cubes has required a compromise between fidelity of representation and processing/storage constraints. Processing the full-resolution data using marching cubes results in an excessive number of polygons to represent and render the model. Even if polygon decimation were used on a full-resolution middle-ear cavity model, an optimistic estimate of 90% polygon reduction would still correspond to a polygon database of 50k polygons. With sufficient hardware, this may not be an unmanageable number, but it is certainly beyond the capabilities of the low-end hardware we wish to support. By reducing the resolution by four in each dimension, we have achieved the desired polygon database size, but in doing so, the accuracy of the model has been reduced. The addition of the middle-ear cavity to our work is useful despite this limitation, however, as it supplies an important macro-structure whose presence assists in spatially orientating the user.

He *et al.* describe a technique that attempts to address the problem of accurately representing the surface contour of an volume-sampled object that has been subject to resolution reduction [He *et al.* 1995]. They produce multiple surfaces in the vicinity of the surface of interest, using a range of marching cubes thresholds that bracket the surface transition. The inner surfaces are then rendered with greater opacity than the outer surfaces

(i.e., a ramp opacity transfer function) creating a gentler surface transition. This approach performs object-space anti-aliasing and produces superior rendered results than the box-car filter we used, with its single hard threshold. The resulting number of polygons in the model will be approximately proportional to the number of additional surfaces used. He *et al.* report good results using five transition surfaces. This approach may be a useful route to increasing the accuracy of the middle-ear model (our work already includes the required polygon sorting for transparent rendering, described in Chapter 6) but it is not clear whether the added benefit would justify the significant increase in the polygon count. This is an area of refinement that could be further explored.

5. VISUALIZING ANATOMY WITH VRML

5.1 Introduction

The VRML file format has encouraged open standards for sharing 3-D models on the WWW. This makes it particularly attractive for use in medical education, where access and portability are important criteria for acceptance of new technologies in the curriculum. This chapter positions the evolving VRML standard amongst other existing 3-D formats, and presents an overview of some of the available VRML browsers. Finally, VRML scene creation is discussed, and the VRML conversion of the 3-D anatomical models described in the previous chapter is presented.

5.2 3-D browsers and file formats

VRML is just one of the numerous file formats now available for sharing 3-D data. Often, the file format is inextricably linked to a particular software and/or hardware environment. Commercial visualization software (such as *AVS*, *IBM DataExplorer*, *Iris Explorer*, and *Wavefront DataVisualizer*) have provided proprietary solutions for the demanding tasks of scientific visualization and the entertainment industry. These tools require workstation power, typically from a restricted set of UNIX platforms, and possession of the costly software for browsing or contributing to a 3-D project. Recent trends toward open standards for graphics environments, however, have promoted the development of software visualization tools that use standard 3-D APIs (Application Progamming Interfaces) to completely abstract the graphics-hardware system layer. The multi-platform support of OpenGL underlies several such software environments, including Silicon Graphic's *Open Inventor*, and the public-domain *Visualization Toolkit.* These both provide object-oriented C++ class libraries for representing and processing data for 3-D display, and can provide the foundation for the development of custom browsers.

A specification for sharing virtual-reality worlds has been difficult to achieve; this is to some extent due to the incompatibilities of hardware I/O devices in various systems, but it is

also a reflection of the diversity of directions and goals existing within the VR community itself. Some specifications, based on elements of VR, have been designed to target specific problems. Apple's *QuicktimeVR* and Omniview's *PhotoBubble*, for example, are file formats and associated browsers that allow the creation of artificial worlds from a series of photographic images. A mosaic of images can be 'stitched' together and sheared to form 360-degree views (called *panoramas*) from a stationary viewpoint. While this approach restricts interaction and limits the types of scenes that can be created, the simplicity of construction (very little scene description is required once adequate scene coverage is provided), and the impressive visual impact that is possible with real-world photography, make this approach appealing for areas such as architectural 'walk-throughs' of museums and real-estate properties.

5.3 VRML description

Some of the obstacles to the creation of a open and flexible standard for the exchange of 3-D scenes on the WWW were addressed with the release of the VRML 1.0 specification [Bell *et al.* 1994]. While it certainly did not satisfy all the needs of the VR community, it was instrumental in starting the process of widespread VR accessibility, while avoiding any restrictive hardware or software dependencies. The specification, a subset of the *Open Inventor* format, defines the geometries and reflective properties of 3-D polygonal models in a hierarchical manner. The hierarchical structure allows objects to be conveniently grouped by their inter-object relationships and common characteristics. Transformation nodes, for example, allow objects to be positioned relative to other objects, rather than by their absolute (world) coordinates. Lighting values and camera positions can also be defined to specify viewing conditions for the models.

VRML syntax can describe various properties of polygons including material properties, normals, and *texture maps.* Material properties include surface colour (ambient, diffuse, spectral and emissive components), as well as transparency and shininess. Normals are used to perform realistic lighting calculations, allowing smoothly shaded surfaces to be rendered. *Texture mapping*, of particular importance for our work, is a method of providing

detail to rendered surfaces without excessive computation. In order to view section images in their correct orientation, to use the example of our work, a rectangular polygon geometry can be used to define the section's 3-D position, and the image itself (consisting of colour and possibly transparency components) can be 'pasted' on the polygon by the process of texture mapping. The images or textures are appropriately sheared according to the viewing transformation, so that they are correctly positioned within their polygon border. While texture mapping does substantially reduce rendering effort, the required calculation (typically provided by the graphical environment) still tends to be quite time-intensive: it is often delegated to hardware in advanced systems.

The interactivity of VRML 1.0 implementations is limited to navigating static models: users can rotate and translate models, or 'fly' to regions of interest. This is a first step towards interaction involving dynamic models in which, for example, planar cuts can be performed by the user, or portions of the model can be individually manipulated. The ability to incorporate portable programs that modify the scene (via local *scripts* written in languages such as Perl, Java, and Tcl) is now available with VRML 2.0. Other additions, such as support for localized audio (i.e., the sound intensity varies with the viewpoint's position with respect to the audio source) and video (via *MovieTextures* nodes), will also enable VRML 2.0 to be an effective multimedia tool in the future. This specification was released in August 1996 and is now under review for certification as an ISO standard.

5.4 VRML browsers

Since the release of VRML 1.0, numerous VRML browsers have become freely available on UNIX, PC and Macintosh platforms. The VRML Repository [SDSC 1996] currently lists 33 different browsers, and details their corresponding platform and feature support.

5.4.1 Multi-platform support

The benefits of multi-platform support for VRML browsers are persuasive: it increases accessibility among users working on a diversity of platforms, it encourages open standards among developers, and it simplifies the design of load-sharing in heterogenous systems where, for example, servers can optionally process data for less powerful client machines.

The extent to which a browser can be efficiently made available on multiple platforms is significantly dependent on the underlying graphics API and associated rendering engine that is in place. (The other main obstacle is user interface management, which is operating-system dependent; Sun's *Java* programming language promises to address this problem.) Although many APIs exist, only a few have gained wide general acceptance and use in VRML development. Some of the most widely used APIs will be described below.

PHIGS [ANSI 1988] was one of the pioneering 3-D graphics standards, and while it is not used by any current VRML implementation, it remains a reference point for many of the more recent APIs due to its extensive range of features. Its use remains largely within the realm of workstation environments due to its size, complexity and need for hardware acceleration to achieve adequate performance. As well, its restricted support for parameter passing makes it difficult to efficiently implement multiple object instances that have slightly varying characteristics. *OpenGL* [Neider *et al.* 1993], originating from Graphics Library (GL), the Silicon Graphics IRIS 3-D library, is one of the few APIs to support all major platforms (UNIX, PC and Macintosh), and has extensive low-level rendering features. Many manufacturers offer graphics accelerators that implement parts of OpenGL at the hardware level. Sun's *Java*-based 3-D library has not yet materialized, but promises the multi-platform support that Java provides. Apple's object-oriented API *QuickDraw3D* [Apple 1996] is available for Macintosh and PC platforms and has extensive support for complex geometrical primitives such as non-uniform rational B-splines (NURBS). Several APIs target PC environments: among them are Intel's *3DR* [Intel 1995], which is optimized for the Intel 486 and Pentium processors, and Microsoft's *Direct3D* [Microsoft 1996]. *Direct3D* is a multi-layered, object-oriented architecture that permits flexibility in the implementation of low-level rendering. *OpenGL*, for example, can be placed at the lowest software level, giving access to the extensive graphics hardware that now supports that standard. *Direct3D* can provide some of the quickest software-only implementations on the PC, and a considerable number of PC graphics-hardware designs have begun to support much of the *Direct3D* API in hardware.

The vast majority of browsers support the PC/Windows platforms, using *OpenGL*, *Direct3D* or proprietary rendering engines. Few VRML browsers, including those that use *OpenGL*, have concentrated on full cross-platform support. *VRweb* targets UNIX, PC and Macintosh platforms using *OpenGL* [Pichler *et al.* 1995], and is one of only two browsers available for experimentation and research in source-code form. The other is SDSC's *WebView*, available for SGI platforms only. The *VRweb* designers hope to maintain this multi-platform support in their VRML 2.0 browser by rewriting their current browser in *Java*, including the use of the *Java* 3-D library. *VRweb* is a cooperative effort between groups at the Graz University of Technology, the National Center for Supercomputing Applications (NCSA), and the University of Minnesota.

5.4.2 Rendering features

VRML browsers vary considerably in terms of rendering capabilities. *OpenGL*-based browsers tend to give higher-quality results due to the ability to control, at the low level, its sophisticated rendering pipeline. Texture mapping is supported by most browsers, but not all do so with predictable results [Reed-Ballreich 1996]. Material and texture transparency is supported to a much lesser degree, due to implementational complexity. Those that do support transparency tend to use the fast but low-quality *screen-door* approach (to be explained in the next chapter on transparency). Examples of browsers taking this approach include Netscape's *Live3D*, SGI's *WebSpace*, Virtus' *Voyager*, Integrated Data System's *V·Realm*, and Chaco Communications' *VR Scout*. The *VRweb* project released in September 1996, subsequent to the development of the present work, a high-quality transparent mode based on the Binary Space Partitioning (BSP) algorithm, as an optional feature within its browser, but it is only available in their UNIX release; neither Windows nor Macintosh support is yet available. To achieve smooth shading from VRML models having only their geometry specified, automatic normal generation is supplied by most browsers; *VRweb*, however, only supported this in their September 1996 release. Normally, smooth shading is implemented using the Gouraud algorithm; Vream's *WIRL* is one of the few browsers supporting the higher-quality but more expensive Phong algorithm.

5.4.3 Dynamic models

Few browsers have implemented the recently finalized VRML 2.0 specification. A small number of early implementations have appeared for Windows platforms (including SGI's *Cosmo Player*, Sony's *Community Place*, and Real Spaces' *RealVR Traveller*). Since many proposals were considered for VRML 2.0, numerous browsers now incorporate their own methodologies for dynamic models (or *behaviours*); with time most of these temporary solutions will migrate to VRML 2.0.

5.4.4 Other features

Most VRML browsers can act as stand-alone applications that are called upon by the Web browser (Mosaic, Netscape, Internet Explorer, etc.) as a *helper* application when a VRML file is accessed. Many are also available as *plug-ins* which integrate into the Web browser itself: a VRML scene appears within the browser window, similar to the way browsers display 2-D images embedded within HTML files.

Collision detection is a feature offered by some browsers that, as the name suggests, prevents the viewpoint from passing through models, and forces models to collide with each other upon intersection. Doors must be opened before entering a building, for example. While this feature increases realism, it requires significant overhead during rendering, and consequently degrades interactive response times. Collision detection has been neglected by the majority of VRML 1.0 browsers, where the worlds are static, but with the dynamics of VRML 2.0 this feature will become more important.

Multi-user worlds provide the ability to interact with other users in the virtual environment. Each user is represented by an *avatar* in the 3-D space, and avatars can engage in discussions (via keyboard or audio) with each other. This is an extension of the text-based MUD (multi-user Dungeon) environments that have been popular on the Internet; such environments are particularly well suited to interactive gaming. VRML developers supporting the browsers and servers required for multi-user capability include Black Sun's *CyberGate* and *CyberHub Client* and Chaco Communications' *Pueblo* (both PC-based).

5.5 Building the VRML scene

5.5.1 VRML construction tools

Although in our work the VRML scenes were constructed by text-oriented manipulation (see section 5.5.2 below), other creation tools now exist (such as Integrated Data System's *V·Realm* for the PC) that allow the scene to be constructed in a 3-D environment and ease the effort required to manage large databases of objects. Objects can be picked and dragged for correct positioning within the 3-D scene. The scene graph showing the hierarchy of object and properties can be easily manipulated with these tools, simplifying operations such as the insertion and deletion of objects. In addition, optimizations are provided to reduce VRML file sizes and increase rendering efficiency. Automatic generation of *Level-of-Detail* (LOD) nodes varies the degree of detail describing an object, depending on its distance from the viewer. In this way, objects projecting onto small areas of the viewing plane do not require wasteful rendering effort. Other efficiency enhancements include maximizing the use of primitives such as cubes and spheres during object modelling, and facilitating VRML *instancing,* where objects occurring multiple times in the database are defined only once. Other classes of tools, such as modelling and animation packages, will become increasingly useful as VRML 2.0 emerges. It should be noted that these construction tools, unlike their browser counterparts, tend to specifically target the PC platform, and are not generally distributed as free- or share-ware products.

5.5.2 VRML translation of the models

VRML conversion from the three data formats of the models described in Chapter 4 involved changing the data representation used to describe the triangle meshes, and adding the image array. VRML *IndexedFace* nodes consist of an array of vertex coordinates, followed by an array of connectivity indices defining the polygons. The marching cubes output was in *vtk* format and already used this form of data representation, making the translation a simple change of syntax. The contour-based models represented the mesh as a series of triangle-vertex coordinates. Translation therefore required identifying the minimal set of vertices and constructing the table of connectivity indices. This task was simplified by the use of associative arrays, available with the Perl scripting language. The images were

incorporated as VRML inline images (*Texture2 image* nodes), an ASCII text representation that is wasteful in terms of file size, but was the only format available with the *VRweb* browser.

Once the translation was complete for the three data paths, each dataset was registered with respect to the others. This was necessary due to variations in scaling, and the lack of a common reference for translations and rotations along the various processing paths. The contour-based 3-D models were first manually aligned with the registered section images. This was done manually by selecting two slices close to the extents of the malleus and tympanic membrane models, and iteratively adjusting scaling, rotation, and translation parameters until an approximate best fit was found for the two slices. Figure 5-1 shows the success of the alignment for these two slices. The residual error for rotation was on the order of the rotational error for the inter-slice registration (3°), and was considered acceptable.

Figure 5-1 Alignment of 3-D models and section images for two sections 68 slices apart. The estimate on the rotation misalignment is 3 degrees.

The registration transformation for the section images and the region-based 3-D model (middle-ear cavity) was known a priori, since the latter was the data source for the former, and all intervening transformations were known. Figure 5-2 shows a rendering of a section and the model after registration. It can be seen that locus of intersection of the cavity model with the image approximates the contour of the section. As described earlier in Chapter 4,

the loss of resolution and averaging that occurred during the volume-data filtering and decimation stage reduced the correspondence of the marching-cubes-generated surface and the original cavity contour.

Figure 5-2: Integration of the VRML middle-ear-cavity model, showing a visible surface view. The degree of correspondence between the intersection of the middle ear cavity model and the cavity contour in the section is an indication of the fidelity of the model to the source data.

6. ADDING TRANSPARENCY

6.1 Introduction

So far we have described the processing of the histological data into 3-D models. Adequate display of these models is another critical problem to address. In order to better appreciate 3-D scenes on a 2-D graphical display, visualization enhancements such as stereo viewing, shading, shadowing, depth cueing and transparency are some of the possible approaches. This section will look at the last of these techniques and describe an implementation, within the VRML browser *VRweb*, that has been created for improved 3-D viewing of polygonal models.

6.2 Motivation

As noted in the previous chapter, there is not widespread support for transparency in current VRML implementations. There is even less support for transparent textures. For this reason, the *VRweb* browser was modified in our work using the Binary Space Partitioning (BSP) approach.

As transparency is part of the VRML language specification, authors of VMRL models can specify that certain surfaces be transparent in order to achieve certain effects not possible when a visible-surface or wireframe rendering mode is being used. Transparency allows one to see all surfaces, providing a means for visualizing large amounts of data simultaneously. It can also be used to cut away surfaces by specifying that certain regions be completely transparent, in order to reveal interior structures. (When transparent texture mapping is available, this can be implemented with a technique called *texture thresholding* [Schroeder *et al.* 1996].)

In the particular case of our work where the focus is on anatomical models, it is anticipated that using transparency will improve the contextual perception of serial sections among their corresponding 3-D structures. This will be possible by displaying the sections as

transparent texture maps and the 3-D structures as polygonal surfaces with transparent material properties. It is hoped that the depth relationships of structures and sections will be apparent with a proper implementation of transparency, enhancing the perceptions possible with normal visible-surface viewing. Other model parameters, such as the choice of screen and colour resolutions, will also influence the degree to which transparency aids visualization.

The ability to create transparent views is usually dependent on solving the more general problem of visibility priority. If the correct depth order of polygons can be determined, the *visible-surface* problem can be solved as well. The only difference between transparent and visible-surface renderings is that the former requires a blending operation during the frame buffer write while the latter requires a simple overwrite. Thus, implementing transparency for VRML solves both problems at the same time.

Having an efficient method for determining visibility priority is advantageous both on machines with specialized graphics hardware (such as hardware z-buffers) and on those that primarily rely on software rendering. Software z-buffers tend to be rather slow since they rely on depth comparisons for every pixel rendered: more efficient methods such as BSP trees can allow much higher frame rates. Even when hardware z-buffers are present, transparency algorithms do not normally take advantage of them. If the graphics sub-system integrates hardware polygon scan converters, even more benefits can be obtained, since these systems can be significantly faster than z-buffered hardware [Foley and Van Dam, 1990]. Having efficient visibility algorithms, therefore, can be useful with many current graphics environments.

6.3 Transparency: image blending

Reflective surfaces can exhibit specular or diffuse reflection. Similarly, surfaces that transmit light can do so without distortion (transparent) or in a diffuse manner (translucent). One can see through transparent materials, although in general the rays are bent by refraction at the surfaces. Non-refractive transparency ignores any such ray bending and is the simplest to implement. When refraction is ignored, everything in the line of sight

through a transparent surface is also geometrically in the line of sight: there is no distortion [Foley and Van Dam 1990]. The non-refractive case will be considered here, since the reduced complexity of line-of-sight viewing is probably a more appropriate goal for visualizing anatomy.

When one object is seen through another, it is necessary to approximate how the colours from the two objects blend. If a transparent polygon 1 is located between the viewer and an opaque polygon 2, the intensity of the individual color components of their projection can be calculated by linear interpolation of the individual intensity components:

$$I_\lambda = (1-k)I_{\lambda 1} + kI_{\lambda 2} \tag{6-1}$$

where the transmission coefficient k measures the transparency of polygon 1, and ranges between 0 and 1. When k is 0, the polygon is opaque and transmits no light. When k is 1, the polygon is completely transparent, and contributes nothing to the intensity I_λ. If k varies with wavelength (making k_λ a more appropriate designator), the surface of polygon 1 selectively passes impinging light, and thus behaves as a colour filter. This is referred to as *filtered transparency*. Only polygons having a constant k across their surface will be considered in the subsequent discussion. The inverse term (1-k), referred to as the opacity or alpha value, instead measures the amount of reflected light. Eqn. (6-1) is thus often termed *alpha blending*. Another way of interpreting opacity is to consider it as the *coverage* of the pixel. If one imagines that only a certain amount of a pixel's area possesses colour while the remaining area remains transparent, the fraction of the area covered can be considered the opacity.

If multiple polygons are in the line of sight, the rendering is most easily done back-to-front using eqn. (6-1) recursively, using the previously calculated I_λ as $I_{\lambda 2}$. For this reason it is normally necessary that an algorithm sort the polygons according to their z-coordinates.

Most graphical environments include drawing modes that implement interpolated transparency using four channels for the colour and alpha components RGBA. Often, the

graphics hardware stores the RGB values in the *frame buffer* for display; when another hardware buffer is available for the alpha component (called an *alpha buffer*), the transparency calculations described below can be done significantly more quickly.

To render each pixel, eqn. (6-1) is calculated for each channel, using the alpha value for the opacity (1-*k*). For example, the red components are combined as follows:

$$R = \alpha_1 R_1 + (1-\alpha_1) R_2 \tag{6-2}$$

where the blend order is the same as eqn. (6-1). The green and blue components are calculated in the same way. It would be convenient if the alpha component of the composite were also calculated this way:

$$\alpha = \alpha_1 \cdot \alpha_1 + (1-\alpha_1)\alpha_2 \tag{6-3}$$

The fact that this is not quite correct is evident upon consideration of opacity as *coverage*. The coverage can be considered to be the probability of a sub-pixel point being filled with colour. The coverage of a composite of pixels will then be the sum of the two coverages minus the intersection of their coverages:

$$\begin{aligned} \alpha &= \alpha_1 + \alpha_2 - \alpha_1\alpha_2 \\ &= \alpha_1 + (1-\alpha_1)\alpha_2 \end{aligned} \tag{6-4}$$

Because the coverages are in fact single-valued (i.e., one for each pixel), the result of this calculation is an average value; sub-pixel details are not considered. This averaging essentially assumes that the area of the pixel covered is randomly distributed, so that the fraction of one polygon covered by a second (within a particular pixel) is the same as the fraction of the whole pixel covered by the second polygon. The consequence of this assumption in practice is that compositing images with very fine detail that is parallel in the two images can have bad results [Foley and Van Dam 1990].

Graphics environments do not tend to provide asymmetrical calculations for colour and alpha components such as those of eqns. (6-2) and (6-4). One way to ensure symmetry is to pre-multiply image colour components by their alpha value before compositing. Using this idea, Porter and Duff have developed a simpler and more convenient and elegant form for many compositing operations. Using their terminology, the operation for compositing polygon 1 *over* polygon 2 is:

$$\tilde{C} = \tilde{C}_1 + (1 - \alpha_1)\tilde{C}_2 \qquad (6\text{-}5)$$
$$\alpha = \alpha_1 + (1 - \alpha_1)\alpha_2$$

where the tilde notation indicates that the colour component C (R, G, or B) is multiplied by its corresponding alpha value [Porter and Duff 1984, Blinn 1994]. Using this form, the blending of image fragments can be done either back-to-front or front-to-back. Front-to-back blending is convenient, for example, when a composite image is to be placed over an unspecified background at a later time. When blending operations are complete (possibly using multiple images), the final colour components are determined by a division by alpha. This pre-multiplication is not required for our particular implementation since the background onto which all polygons are blended (in back-to-front order) is assumed opaque (α=1), so that the composite alpha value is always 1. The use of eqn. (6-2) is therefore sufficient for the colour components, while the composite alpha value—incorrectly calculated in the symmetrical fashion of eqn. (6-3)—can be ignored.

6.4 Standard approaches

The methods of implementing transparency basically fall into two categories: those that require an ordered rendering of scene polygons, and those that do not. Ordered rendering normally begins with those polygons farthest away from the viewer (those with the most negative z-coordinate in the canonical view-transformed space), and proceeds with polygons progressively closer to the viewer. The list-priority and scan-line algorithms described below take this approach. Other algorithms that effectively sort polygons are possible, but will not be considered in detail. The Weiler-Atherton area-subdivision algorithm [Weiler

and Atherton 1977], for example, requires a complex clipping algorithm that can consider polygons with concavity or holes. Ray-tracing techniques [Appel 1968, Mathematical Applications Group Inc., 1968, Goldstein and Nagel 1971] tend to be prohibitively expensive, although they are well suited to parallel implementation. Other approaches, such as screen-door transparency and z-buffer-based algorithms, are appealing in their simplicity since they allow rendering of polygons in any order. All of these methods can be judged in terms of their relative speed and rendering quality, their implementation complexity, and their applicability to certain environments.

6.4.1 Ordered-rendering approaches

These approaches rely on a scheme that orders the polygons, normally in back-to-front fashion. Rendering can be performed by successive compositing of pixels (as described previously), starting from the polygon farthest from the viewpoint, and proceeding to the nearest polygon. The main source of complexity in these algorithms involves resolving the depth ambiguity when projections from multiple polygons intersect. Polygon splitting is often required to correctly determine a depth-sorted list of polygons. Many visible-surface algorithms may be adapted for this purpose, but those that discard polygon fragments (such as Warnock's area-subdivision algorithm [Warnock 1969] or a front-to-back BSP algorithm [Gordon and Chen 1991]) before scan-conversion are not suitable. Ordered- rendering approaches can be divided into two families: those that split models at *object precision* and those that split at *image* (or *pixel*) *precision*.

Object-precision algorithms have the advantage of displaying models at any resolution (i.e., they operate on the original *continuous* object data) for a given viewing direction, once object visibilities have been determined, and are not limited by the screen-resolution object *sampling* that occurs with *image-precision* approaches. Enlarging a scene image, for example, only requires another display step with object-precision algorithms since the visibility relationships have already been calculated. Image-precision algorithms, however, must recalculate the scene at the new sampling rate before display. Ultimately, of course, some sampling has to occur in either case for rendering on (discrete) raster displays. The

uniform resolution of object-precision approaches can, therefore, produce wasted efforts in certain cases, since small, distant objects may be rendered with detail that far exceeds the support of the final screen resolution.

Two sets of algorithms, representative of these families and modest in resource requirements, will be examined here. List-priority algorithms will be considered for the case of object precision, and scan-line algorithms will considered for the latter case.

6.4.1.1 List-priority

List-priority algorithms begin by creating a visibility ordering of objects. This is done before scan conversion, so that rendering the ordered objects back-to-front (using the *painter's algorithm*, where successively closer surfaces are drawn over previously drawn surfaces) ensures correct results. Splitting of objects in their world coordinates is done to resolve ambiguities such as overlap in z-extents. The most common examples of list-priority algorithms, depth sort and BSP-trees, are described below.

6.4.1.1.1 Depth sort

There are three stages to the depth-sort algorithm [Newell *et al.* 1972]. Polygons are first sorted based on their farthest z-coordinate. Some ambiguity remains, however, for polygons overlapping in z-extent and whose projections intersect. These polygons are split so that no z-overlap remains. Polygons can then be rendered from back to front. This algorithm tends to slow down with greater numbers of polygons for two reasons. Not only is sorting at least an $O(n \log n)$ operation, but the use of more complex splitting cases becomes more common with increased scene complexity [Foley *et al.* 1990]. As well, any change in view position requires that processing restart from the beginning.

6.4.1.1.2 BSP (binary space partitioning) trees

In the BSP-tree method [Fuchs *et al.* 1980], a preprocessing step is done just once for a static scene. Remarkably, this step creates a binary-tree structure which can be used to describe the visibility relationships between polygons from any viewpoint. The rendering stage can then use this preprocessing information to render polygons in linear time ($O(n)$

where n is the number of polygons), again, regardless of viewpoint position [Foley *et al.* 1990]. The potential weakness of the algorithm is the potential for large numbers of polygons to be created by the splitting which occurs during the preprocessing step. There are ways, however, to reduce this problem. A more detailed description of the BSP algorithm will follow in section 6.5.

6.4.1.2 Scan-line algorithms

The scan-line algorithms [Wylie *et al.* 1967, Bouknight 1970, Bouknight *et al.* 1970, Watkins 1970] operate at image precision. An image is created one scan line at a time, and at each scan-line pixel, the depth of each polygon in the line of sight is checked. For rendering visible surfaces only, the closest polygon is the only one that needs to be written. For transparency, the pixel is successively blended from the farthest to the nearest polygon. Various forms of *coherence* can be used to speed up this algorithm. Coherence refers to the degree to which parts of an environment or its projection exhibit local similarities [Foley *et al.* 1990]. If one assumes that none of the polygons penetrates another (this can be enforced by splitting all penetrating polygons), *depth coherence* implies that the relative depths of polygons do not change within the scan line. Depth coherence can also be exploited across successive scan lines, if it is found that the active polygon edges remain constant and in the same order across the scan line.

6.4.2 Unordered-rendering approaches

Much complexity in rendering logic can be reduced if one can avoid the preprocessing stages of the list-priority algorithms, and the multiple depth checks of the scan-line algorithms. The following methods allow rendering to be done without explicit polygon sorting.

6.4.2.1 z-buffer methods

Z-buffers, also known as *depth buffers*, are contiguous areas of memory (either allocated at run-time or dedicated to the graphics sub-system) providing an image-precision method of determining the visible surfaces of an arbitrarily ordered set of polygons. In addition to the standard frame buffer holding pixel colour values, a *z*-buffer system possesses an auxiliary

buffer that holds the z-value of a rendered pixel. Initially, the z-buffer is set to the z-value at the back clipping plane (i.e., the farthest z-value from the viewer). During rendering, a polygon's z-value at each pixel is compared with the current z-buffer value at that pixel, and pixel colour and depth values are replaced if the polygon's z-value is closer to the viewpoint. This process is continued for all scene polygons, with the result that only visible surfaces are displayed. In certain respects, z-buffer methods are similar to scan-line approaches, but only one depth check is required at any stage. A hybrid method has been suggested for visible-surface determination [Myers 1975].

Because the polygons are rendered in random order, however, some modifications of the basic algorithm are required for transparency. Mammen suggests a z-buffer-based system requiring multiple rendering passes to resolve the z-ordering at each pixel [Mammen 1989]. This requires significant intermediate buffers and a depth check with upper and lower bounds. Normally z-buffer implementations check for simple comparison ($z<z_0$ or $z>z_0$) but not for range ($z_0<z<z_1$). This method of implementing transparency is consequently restricted to specialized hardware or software.

6.4.2.2 Screen-door transparency

This method, also referred to as stippling transparency [Neider *et al.* 1993], is unique in the sense that no sorting of any kind is required. The surface of each polygon is rendered as a mesh (called the stippling pattern) whose density is determined by the desired opacity. This approach, while appealing in its simplicity, produces less accurate results since the mesh of any one polygon partially obscures other polygons behind it. In addition, the choice of stippling pattern, which is essentially a dithering issue, affects the results: ordered dithering is less accurate than random dithering when several transparent polygons overlap. This is because ordered dithering uses the low-order bits of each pixel's (x,y) address as an index into an object transparency mask. If the indexed bit is 1, the pixel is written; if not, the next closest polygon will be visible at that pixel. The problem with ordered dithering is that when objects are drawn with the same mask, polygons with more distant z-values are fully obscured. Both dithering methods reduce spatial resolution, and rely on the eyes to perform

spatial integration to achieve a perception of blending [Foley and Van Dam 1990]. Figure 6-6 below compares rendering results using interpolated and screen-door transparency methods.

6.4.3 Rationale for selected approach (BSP trees)

The BSP-tree algorithm is the most suitable for the implementation of our work. The anatomical models to be used are essentially static so that once the BSP preprocessing is done the viewpoint can change without further processing. The algorithm achieves very efficient rendering times with accurate blending results (assuming sufficient colour-depth exists in the graphics hardware). Furthermore, from the perspective of implementation, its relative simplicity and the public-domain availability of core code permit quick development.

6.5 Polygon sorting by BSP (binary space-partitioning) trees

This list-priority algorithm developed by Fuchs, Kedem, and Naylor [Fuchs *et al.* 1980, Fuchs *et al.* 1983] is a very efficient method for calculating the visibility of static 3-D polygons from an arbitrary viewpoint. After possibly time-intensive preprocessing, the display time for this approach is linearly related to the number of polygons [Foley *et al.* 1990]. Normally, scenes must be static for this algorithm to be advantageous, although some researchers have described the use of BSP trees in dynamic scenes [Torres 1990, Chrysanthou 1992]. The BSP algorithm is well suited to the anatomical scenes of our work, which generally are unchanging, except for the position of the selected slice or the addition/removal of whole structures. In these cases, avoiding recalculation of the entire BSP data structure (i.e., allowing partial BSP updates), as described by Torres and Chrysanthou, may be more efficient.

The development of the BSP-tree approach was based on the work of Schumacker [Schumacker *et al.* 1969], who observed that environments can be considered as clusters, or collections of faces. If a plane can be found that separates one set of clusters from another, then clusters that are on the same side of the plane as the eyepoint can obscure, but cannot be obscured by, clusters on the other side [Foley *et al.* 1990]. Each of these clusters can be

recursively subdivided by suitable separating planes. If a cluster spans a partitioning plane, it is simply divided into 2 sub-clusters, one on each side of the plane.

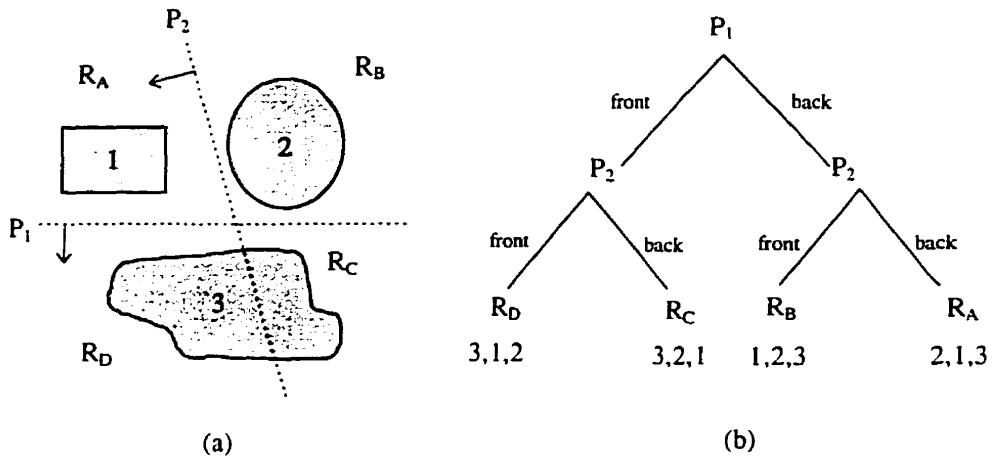


Figure 6-1: Cluster partitioning and priority. (a) Clusters 1,2 and 3 are divided by partitioning planes P_1 and P_2, determining regions R_A through R_D where the viewpoint may be located. (b) Shows the binary-tree representation of (a), with the indicated cluster priorities for each region (from [Sutherland *et al.* 1974]).

This partitioning of the environment can be represented by a binary tree rooted at the first partitioning plane (see Figure 6-1). The tree nodes represent the partitioning planes while the leaves represent regions in space. Each region is associated with a *cluster priority*, the unique order in which clusters can obscure one another if the viewpoint is located in that region. When the clusters are considered to be sets of polygons, the usual case for graphical representations, the spatial subdivision is usually done using the planes of the polygons themselves as partitions (or *autopartitioning* [Paterson and Yao 1990]), and continues until each node contains a single polygon.

With this data structure in place, the correct ordering for scan conversion of the scene from an arbitrary viewpoint is determined by traversal of the tree. The root polygon divides the environment into two half-spaces, with the front half-space indicated by the surface normal. If the viewpoint is in the root polygon's front half-space all polygons in the rear half-space must first be drawn (since they may be obscured by the root), then the root polygon itself,

followed by the front half-space polygons. The reverse is true if the viewpoint is in the rear half-space. This process continues recursively for all the partitioning polygons until all polygons are drawn: each time, the position of the viewpoint relative to the partitioning plane determines the sub-space traversed. Figure 6-2 shows an example environment with its BSP tree, and shows the draw traversal of the tree from two viewpoints.

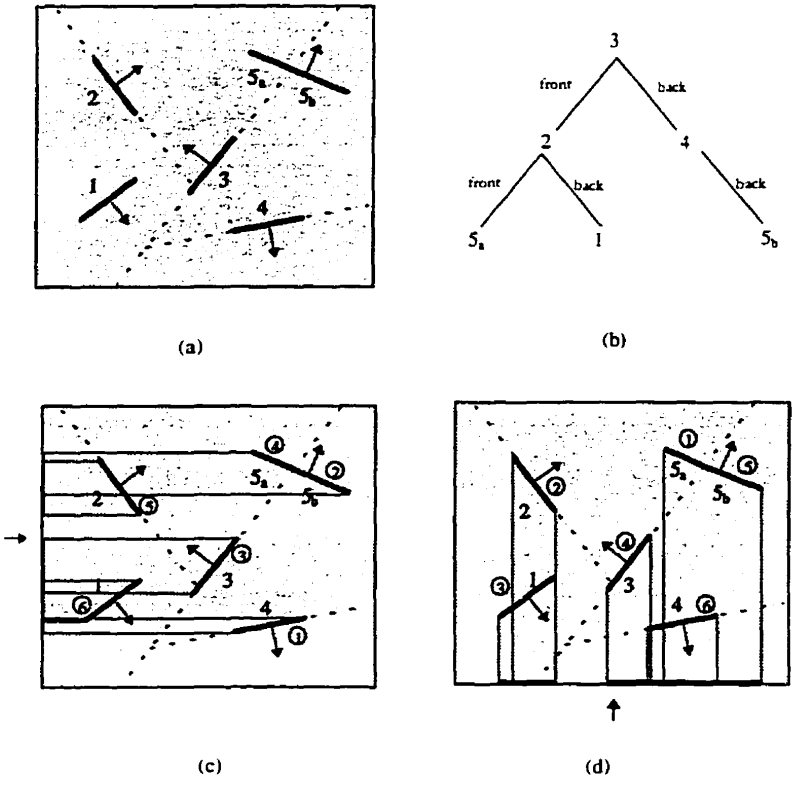


Figure 6-2: Construction of a BSP tree (b) for the polygon scene (a). Polygon 3 is used as the root. Also shown are two traversals of the BSP tree for drawing the scene projection with viewpoint from the left (c) and below(d). The circled numbers indicate drawing order (from [Fuchs *et al.* 1983]).

The polygon selected as the root of each sub-tree can substantially effect the algorithm's performance. The optimal set of roots would be that which causes minimal splitting of the child polygons, since each additional polygon requires more storage and increased rendering time. Finding this set is thought to be an $O(n!)$ problem, but a simple and inexpensive heuristic is as follows: at each level, use a small number of trials (say five or six) to choose

the polygon that splits the smallest number of its children [Fuchs *et al.* 1983]. This will often be a good approximation to the best case. The trial roots can be but need not be chosen at random: it is often effective to merely select the next elements from the input list of polygons. Since the polygon list is often derived from object groupings, this selection strategy tends to choose spatially proximal polygons. Paterson and Yao have described an algorithm that has an upper bound of size $O(n^2)$ (i.e., the worst-case number of resulting polygons after preprocessing n polygons is proportional to n^2) and an upper bound of time $O(n^3)$ [Paterson and Yao 1990]. This is an optimal algorithm with respect to size, since they have shown that the problem of autopartitioning in object-precision visibility algorithms has a lower bound of size $\Omega(n^2)$ They hope to improve the upper bound of time of their algorithm to $O(n^2)$. If the partition selection is not constrained to autopartitions, *free cuts* can be used to reduce this complexity. Free cuts can partition a collection of facets non-trivially without division of any facet (see Figure 6-3(a)). For the normal case where autopartitions are required, Paterson's approach is to preferentially choose *bounded cuts* when possible; otherwise the polygon with the fewest children at each stage is chosen. Bounded cuts are similar to free cuts, but are distinct in that the partitioning facet completely sections a bounded polygonal area (2-D) or an enclosed polyhedral space (3-D). Figure 6-3(b) illustrates a bounded cut for the 2-D case.

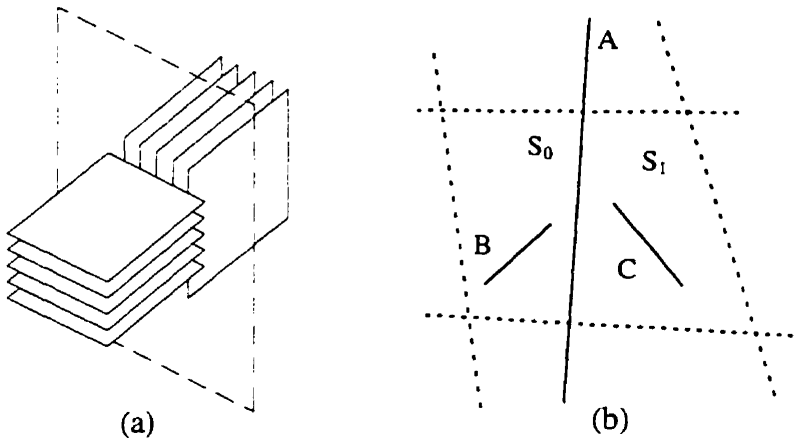


Figure 6-3: (a) An example illustrating the lower bound $\Omega(n^2)$ for autopartitions. Two sets of n squares produce n^2+2n polygons for every autopartition, whereas the outlined free cut reduces this number to $2n$. (b) An example of a bounded cut for the 2-D case. The region S bounded by the four dotted lines is divided by line segment A into sub-regions S_0 and S_1. No extra cuts are created by A within S (unlike other segments such as B or C) [from Paterson and Yao 1990].

He shows that bounded cuts are always advantageous for partition plane selection. Therefore choosing partitions based merely on fewest children is an inadequate strategy, since a bounded cut may create more immediate children while still reducing the overall child count. Detecting and selecting bounded cuts adds significantly more time to the preprocessing, as indicated by Paterson and Yao's upper bound of time $O(n^3)$.

6.6 Methods

In this section we describe an implementation of transparency within the VRML browser *VRweb*. Using the BSP algorithm just described, polygons can effectively be sorted so that back-to-front rendering with alpha blending produces the desired transparent effect.

6.6.1 Software/hardware resources

VRML is simply a file format for hierarchical description of 3-D geometrical relationships, as explained earlier, and does not specify any particular hardware or software for its implementation. This section will briefly describe the environment used for the implementation of our work.

The public-domain VRML browser *VRweb* uses the OpenGL 3-D graphics library (or "work-alikes" such as Mesa [Paul 1993]) as the underlying graphics layer. It can therefore be used on any platform that supports the OpenGL standard, including those that implement parts of the specification in hardware.

6.6.1.1 OpenGL

The OpenGL 3-D graphics library consists primarily of 3-D graphical primitives and of commands for setting OpenGL states such as lighting calculations, surface properties, and view-transformation matrices [Neider *et al.*]. OpenGL can then perform the calculations of the rendering pipeline (view transformation, clipping, shading, etc.) of all defined primitives for a particular viewpoint. OpenGL also includes drawing modes that implement interpolated transparency using four channels for colour and alpha components (RGBA). If the hardware does not support RGBA components (as is the case for many systems having a limited colour palette of 256 colours), an OpenGL implementation can emulate an RGBA buffer in software before colour mapping to the frame-buffer palette occurs. Component coefficients can be chosen for a number of compositing operations, including those that satisfy the blending eqn. (6-2). Texture-mapping calculations are also performed, after definition of the bordering polygon and texture image. For improved performance, several resolutions of the texture can be calculated and stored to provide more detail at close viewpoints and less detail when viewing from afar. When enabled, OpenGL transparently controls the management of this *mip-mapping* process.

Implementation of the OpenGL specification has been made available on a wide variety of platforms. Our work used Windows95 and DEC-Alpha UNIX implementations of OpenGL. These were both software-only implementations: Microsoft's dynamic-link library (DLL) for Windows95 and the public-domain Mesa library for UNIX.

6.6.2 BSP implementation

The core code for the BSP implementation was taken from public-domain sample C++ code [Wade 1994] and modified for use within *VRweb*. It provides base classes for points, polygons and polygon lists that can be used as the input structures when constructing the

BSP tree. This code processes only convex polygons: this is a restriction of the polygon-splitting code, not the BSP algorithm itself. This restriction is quite acceptable for most application, since any concave polygon can be subdivided into convex constituents, if required. (Recall that the anatomical models produced in Chapter 4 are described by triangles.) The enhancements to this software provided by the present work included Fuch's heuristic algorithm for root selection, support for textured polygons, and integration with OpenGL and *Vrweb*. These three enhancements will be discussed below.

6.6.2.1 Partition-plane selection

As mentioned previously, the choice of the partitioning polygon at each node of the BSP tree greatly influences the number of polygons in the resulting tree. In the core BSP code, the selection of the partitioning polygon is essentially random. To improve this situation, we implemented the heuristic suggested by Fuchs [Fuchs 1983] of choosing the partitioning polygon which produces the fewest immediate child polygons. This required a classification function to determine whether a given polygon is in the front or rear half-spaces, or whether it spans the half-spaces and must be split. This classification had previously been embedded in the splitting function: separating the two speeded up the partitioning selection, where no splitting is required. Since testing all polygons at every stage generally has excessive time requirements, a maximum of 6 polygons are tested at each stage (following the suggestion of Fuchs). The actual number tested can be lower if fewer than six polygons remain in the region, or if an acceptance threshold value is reached: if a trial root polygon is found that splits a number of polygons equal to or less than the threshold, that polygon is immediately chosen as the partitioning polygon, with no more testing required. This threshold was set to zero: a reasonable (but not rigorous) approach to reducing the polygon output to a minimum.

6.6.2.2 Textured polygons

To support processing of textured polygons by the BSP algorithm, a *tpolygon* class was derived from the *polygon* class provided by the sample BSP code. It inherits all the properties of polygons (including vertices and material type), but differentiates from the

parent during splitting and drawing procedures. If a textured polygon must be split, both its 2-D *texture coordinates* and its 3-D vertex coordinates must be split. Texture coordinates are normalized values, normally between 0 and 1, that indicate x and y image locations. Texture splitting must be done to obtain the correct image fragment for the corresponding (split) polygon. These two splits are more efficiently done simultaneously, so the texture-splitting logic was merged with the original polygon-splitting logic. The drawing logic requires the extra consideration of texture image and coordinates for textured polygons. Because of this polymorphic design, processing of untextured polygons is not slowed down by texturing logic (other than the overhead imposed by C++ virtual functions, where runtime type-checking is required).

6.6.2.3 VRweb integration

Following the model hierarchy implicit in the VRML specification [Bell *et al.* 1994], *VRweb* creates a scene tree to represent the relationships between objects (following the structure of the public-domain VRML parser QvLib [Strauss 1995]). In this way, for example, scene objects can be grouped in their own local coordinates before being transformed into their world coordinates. Other parameters such as material properties or texture images can be pushed and popped states for certain sub-trees of the scene. When building the BSP tree, the scene tree must be traversed, but subsequent drawing only requires the BSP tree.

Traversal of the BSP tree bypasses the scene hierarchy altogether, since rendering is ordered by polygon visibility, not by object groupings. This "flattening" of the scene tree means that no states can be used in the BSP tree: transformations, textures, and materials all must be associated with every polygon. This increases storage overhead and preprocessing time, but has little effect at draw time since parameters such as materials and textures are sent to OpenGL only when changed: a simple check determines whether new materials or textures need to be set. In this way, *face coherence* can be exploited: proximal polygons will tend to have similar material properties. Similarly, the source of texture images tends to be the

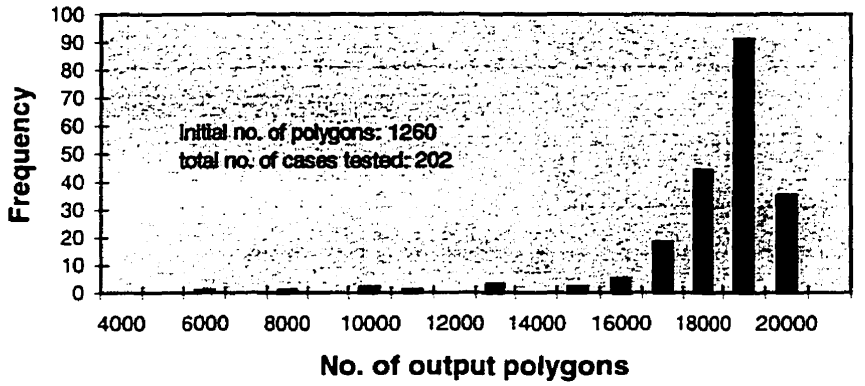


Figure 6-4: Histogram of the number of output polygons using a different BSP-tree root in each case.

same for adjacent polygons. This is especially true for our work, where there is only one texture.

6.7 Results

The performance of the algorithm was tested with respect to preprocessing time, number of output polygons, and rendering speed. The measured times of the tests were based on execution with a 486-66MHz PC without special graphics hardware. Timers were embedded in the code itself, bracketing the procedures of interest.

To ascertain the effectiveness of the root-polygon selection algorithm, it was compared with a strategy of random root selection. Beginning with a sample data set of 1260 polygons, 202 different root polygons were chosen in trial cases, with the subsequent partitioning polygons chosen as they appeared on the input polygon list. The large majority of the cases produced in excess of 16000 polygons, with a mean value of 17850 polygons (see Figure 6-4). The best case found was a total of 5770 polygons.

One of these partitioning cases was chosen for comparison with the results of the root selection algorithm. A typical case of 17986 polygons (a factor of 14.3 increase in the number of polygons) required 223s of preprocessing and took 5.5s to traverse the tree during drawing (in visible-surface mode with neither alpha blending nor texture mapping). When the root selection algorithm with 6 trials was used however, the total number of

polygons produced was 3058 (a factor of 2.4 increase with respect to the initial 1260) in 66s of preprocessing. Draw traversal for this case required 1.4s. These latter figures all demonstrate that significant improvements can occur with a root selection scheme in place.

The same measurements were done for two other models (having 703 and 1569 initial polygons, respectively), and the results are shown in Figure 6-5. These figures also reflect the sensitivity to the root searching parameter: the maximum number of trial polygons at each stage was varied with values of 1 (no searching), 6 and 18. The value of 6 consistently shows a marked decrease in output polygons compared to the no-search case (Figure 6-5(a)). However, the fact that longer preprocessing (with a value of 18) only marginally decreased the number of polygons (and in one case actually increased the number) indicates that Fuchs' suggestion of 5 or 6 test polygons is probably a reasonable value for the heuristic. Figure 6-5(c) indicates the linear dependence of draw traversal time with number of polygons, as expected. It is interesting to note that in both cases using a root selection scheme, the reduction in polygons was so pronounced that not only were the drawing times decreased, but the preprocessing times as well (Figure 6-5(b)). Note that these values were for a few models only and are not the result of extensive testing. It would be interesting, for example, to further observe the trend of the output-to-input ratio for greater numbers of polygons and for other numbers of trial polygons. Nevertheless these results are very likely representative values, and are useful in observing the importance of selecting good partitioning planes.

Images rendered with this BSP implementation within the *VRweb* browser are shown in Figure 6-6. Figure 6-6(a) shows a textured test case rendered with visible surfaces and using interpolated transparency. It can be seen that transparent views look 'washed out' due to blending with grey background: the reduced colour saturation increases the importance of 24-bit colour support in the graphics hardware, to detect subtle gradations of colour. Figure 6-6(b) shows a scene rendered transparently from various angles, and is compared with a transparent scene rendered by the less-expensive screen-door algorithm (using Netscape's *Live-3D* software) in Figure 6-6(c). The lower quality 'meshing' effect is apparent.

One rendering problem appears when larger models are viewed. The colours of certain regions are incorrect: they appear to take on the material properties of neighbouring structures. This error, which varies with the viewing position, is very likely a bug in in the handling of material properties in the BSP code and should be investigated. It can be seen in Figure 6-6(a), where the background portion of the textured slice (which, presumably, has been split by a partition plane) is darker than the foreground.

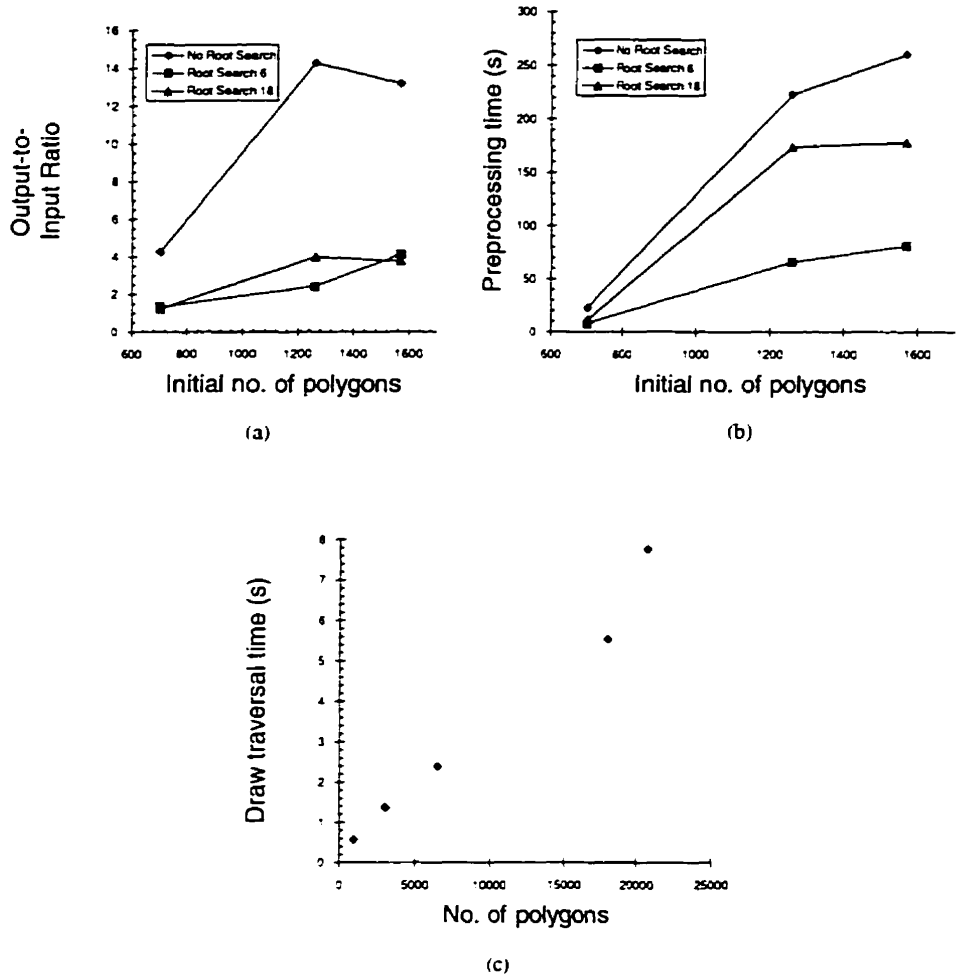
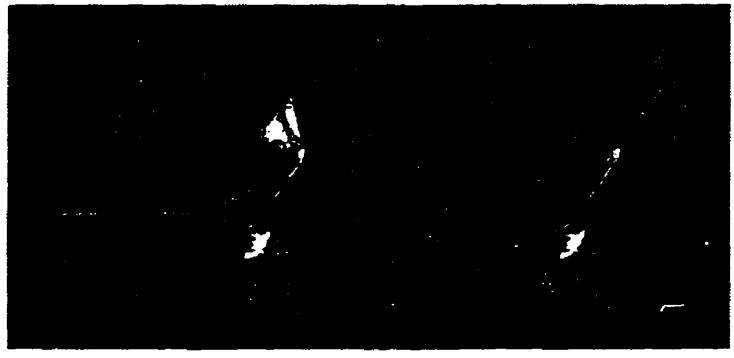


Figure 6-5: Results of BSP processing. (a) Comparison of the ratio of the number of output polygons to the number of input polygons with and without a root searching strategy. Results are for three different models. Note that although the root search for one set was 18, the results were worse than for 6, illustrating the stochastic nature of the partition selection process. (b) Comparison of preprocessing times with and without a root searching strategy. Results are for three different models. (c) Times to traverse the BSP tree during drawing. The linear time dependence is apparent.



(a)

(b)

(c)

Figure 6-6 Sample 3-D scenes: (a) Left: visible surfaces viewing. Right: transparent viewing with opacity values of 0.5 for the tympanic membrane (blue), 0.9 for the malleus (yellow), and 1.0 for the slice. The dark regions in each case are rendering artifacts. (b) Three transparent scenes of the malleus (white) and tympanic membrane (flesh) from different angles. Opacity values same as (a) for the two structures. (c) Examples of screen-door transparency.

6.8 Discussion

6.8.1 Interactive rates

An important question for the usability of this implementation is whether the interactivity times are reasonable when texture mapping and transparency modes are enabled. On the slowest hardware tested (486-66 MHz PC, without special graphics hardware), the refresh rate was approximately 0.1 Hz while a faster machine (Pentium 133 MHz PC, hardware accelerated polygon scan-conversion, hardware double buffering) refreshed at 4 Hz, for a reasonably detailed model (1200 polygons). Even at these slow rates, the user has some sense of navigating 3-D space, but are they are far below what is desired for the immersive goals of VR. Nothing should prevent the use of this data in hardware systems with higher graphics throughput, however. It is interesting to note that at the time of this writing, hardware texture mapping has just entered the mainstream PC graphics market, indicating that the required hardware enhancements will soon be widely available.

6.8.2 Improving BSP efficiency

For the problem of visible-surface determination (but not for transparent rendering) it might be imagined that, for a given viewpoint, parts of the tree are not visible, and can somehow be avoided during rendering. The algorithm of Gordon and Chen traverses a completely sorted BSP-tree front-to-back, and avoids rendering pixels more than once using a special data structure for the screen image [Gordon and Chen 1991]. All nodes are visited during the traversal: time is saved (with respect to standard back-to-front BSP rendering) by reducing the total number of pixels rendered. The more expensive rendering is, the more time is saved with this approach, especially for larger numbers of polygons. When a inexpensive (constant) shading method was used, this front-to-back method rendered in about 30% of the time that back-to-front required, for 5-25k polygons. This trend appeared to continue for greater numbers. When interpolation shading (probably Gouraud) was done, however, the effect was more pronounced: front-to-back took 8% of the time for 10k polygons, and leveled off for larger numbers of polygons, while back-to-front times kept climbing. These savings are due to the elimination of runs of hidden pixels during a scan-

line rendering phase. For more complex scenes, there are greater numbers of hidden pixels and potentially greater savings.

But if we cannot discard polygon fragments, as is the case with transparency, other approaches are required. One possible area of improvement that deserves consideration is whether there is any advantage to stopping the BSP construction before the nodes have single-leafed children, and leaving some sorting to the drawing stage. Such a strategy might, for example, be more adaptable to efficient rendering of changing models.

Since the position of the viewpoint and the orientation of the partitioning planes will be arbitrary, some sorting of all the polygons will be necessary at some point (either during preprocessing or during drawing) in order to determine the correct surface order. The drawing time is determined by the time to visit all the polygons, perform the front/back check with respect to the viewpoint for each node visited, and wait for the graphics subsystem to scan convert each polygon. Would significantly more time be introduced by delaying some sorting until draw time?

Looking at the previous results, it should be clear that the sorting operations are quite expensive with respect to the drawing process. For one case of 1260 original polygons, 66s of preprocessing was required with 1.4s of drawing time. Now if we imagine, for simplicity, that we have a BSP tree that is of uniform depth throughout of n levels, then at the i-th level there are 2^{i-1} nodes. This means there are 2^{n-1} leaves at the bottom level. The total number of nodes for the tree is 2^n-1 (by summing the geometric series). If we postpone even the final level of sorting, we are leaving at least one-half of the nodes unsorted. Obviously, not all BSPs are of uniform depth, but it might seem, unless the tree is severely unbalanced, that a large bulk of the sorting time is spent on the lowest levels of the tree, if we assume that construction times are uniform at all levels of the tree.

But the probability of splitting, a relatively costly event, should decrease as the partitioned spaces have fewer polygons. Thus there should be less splitting, and consequently quicker construction times, at the bottom of the tree. If we assume again that the last level of the tree

remains unsorted during preprocessing, there are potentially 3 polygons (a node polygon and its two children) unsorted with respect to each other and with respect to the viewpoint. Rendering these three polygons may require some sort of splitting to deal with cyclical z-overlap [Foley *et al.* 1990] if a list-priority algorithm such as BSP is used; another resort is to rendering the partitioned space with a different approach altogether. Depth sort, for example, has very good performance for small numbers of polygons (where the expensive z-overlap case is rare) when compared with z-buffer, scan-line and Warnock algorithms [Foley *et al.* 1990]. In addition, it may be more efficient than a process of BSP construction and rendering, which can split even when there is no cyclical z-overlap. Further testing is required to determine the degree of such preprocessing-display load balancing that reaps benefit (i.e., how much BSP-tree construction must be done during preprocessing) and how the rendering phase might best be implemented.

Another interesting extension to our work would be to add the further sophistication of bounded cuts to the root selection and determine the degree of benefit (increased preprocessing time vs. reduction in total polygons) using sample anatomical models. Implementing Paterson's full algorithm would likely lead to excessive preprocessing times with its $O(n^3)$ time bound. However, using the textured slice to be the root polygon (creating a bounded cut) would be a straightforward enhancement. It might also be useful to take further advantage of the layered nature of meshes that are derived from serial-section or voxel data by using all the section planes as the partioning planes. These ideas were not implemented because there is no way to add this information to the VRML code: it would have to be arbitrarily added to the browser logic in a non-general way.

7. CUSTOM SCENES ON THE WWW

The tools and developments described so far allow for navigation of static models: users can rotate or translate models, or 'fly' to regions of interest. In order to allow the user to modify the scene in any way, it is necessary to incorporate some sort of interaction mechanism that augments the basic functions of VRML within the WWW environment.

The scene could conceivably be altered in several ways. The histological section could be selected from a set of images to allow comparison of slice structures as one moves along the slicing plane. Given a proper resampling calculation, the slicing angle could be changed to gain a different perspective on the structures as one moves in a different plane. Showing multiple slices (on possibly different slicing planes) may be useful to increase the spatial orientation of a scene. Optional labelling of structures is useful during the student's learning or review sessions. The colour and degree of transparency could be a parameter that is adjustable by the user to highlight or diminish certain structures. Other modifiable parameters could certainly be envisioned.

In our work, we have focused on altering what seem to be the most basic and easily implemented of the above parameters: slice selection and color/transparency preference. It was hoped that labelling could also be considered, but the VRML *AsciiText* node was not supported by the *VRweb* browser at the time of this development.

The dataset, consisting of the structural models and 130 section images, is too large to be completely transferred to a client over low-bandwidth channels. Since there are 130 sections to choose from, each one having a size of approximately 100 kbytes, it is necessary to provide some means of selecting the slice to view. Being able to choose the colour and degree of transparency may be useful to increase the visibility and clarity of individual structures as well as the entire scene. It was helpful even for testing purposes, since the most

useful combinations were not obvious: a trial-and-error process was simplified when these values were easily modified.

In order for the client to specify these parameters, some mechanism is required to create graphical user interface components such as pull-down menus, radio buttons, etc. If VRML 2.0 were available, some of the scene modification could be done locally on the client machine, using a scripting tool (based on languages such as Perl, Java, or Tcl). This would allow the structure colours and transparency values to be selected and modified without any server interaction. The desired slice could be selected similarly, but a server request would be required to release the file of the slice image. The client would then be able to modify the slice node of the model locally without requiring a duplicate download of the 3-D structures.

Since VRML 2.0 was not yet available at the time of this development, we chose an approach that is similar to the above, but uses an HTML form to submit scene change requests from within a Web browser. HTML forms provide the graphical user interface components as well as a protocol for delivering the form requests to the server. Figure 7-1 shows the screen layout of the form. A Common Gateway Interface (CGI) script residing on the server machine uses the query to create a VRML scene that is sent back to the client's VRML browser for viewing. The major drawback of this approach, however, is that it requires that the entire scene file be downloaded for any scene modification. Figure 7-2 illustrates this HTML-CGI client-server request mechanism.

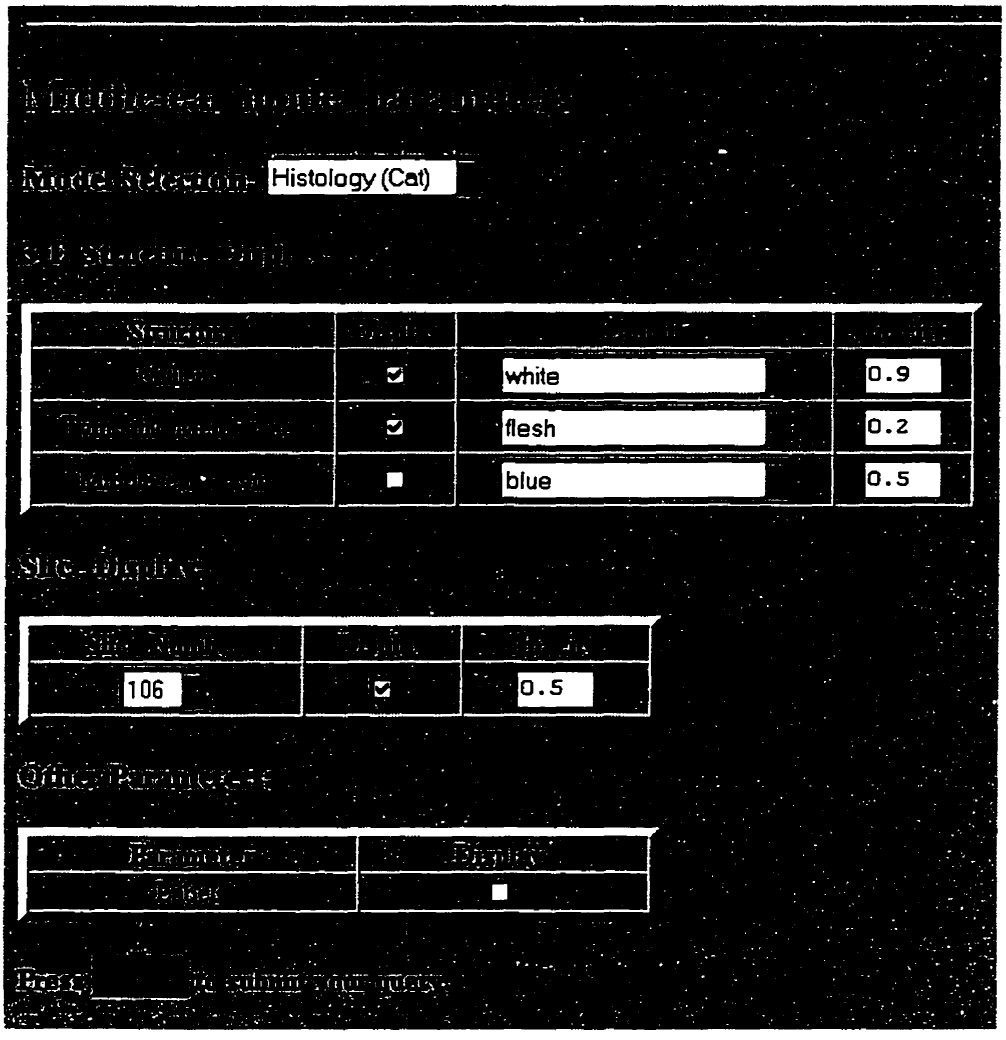


Figure 7-1: Layout of the HTML form for requesting custom VRML scene on the WWW.

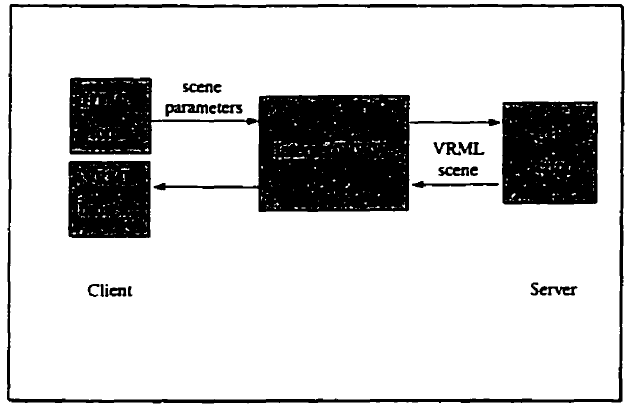


Figure 7-2: Custom scenes on the WWW. Scene parameter selection is initiated by a form submission on the client HTML browser. A Common Gateway Interface (CGI) script running on the server creates a VRML file that is sent back for viewing on the client VRML browser.

8. CONCLUSIONS

8.1 Summary

The use of visualization and virtual-reality techniques in medical education has been described. One of the chief benefits promised by these evolving technologies is to promote anatomical reasoning: by making doctors more aware of the 3-D spatial relationships among anatomical structures, their diagnostic abilities should improve. These tools offer more flexibility than is possible with conventional pedagogical approaches, but it remains to be seen whether they will supplant or enhance standard procedures such as cadaver dissection. It is clear that some of the elements of 'hands-on' exploration will be difficult to simulate for the foreseeable future.

The creation of 3-D models from histological sections of the malleus, tympanic membrane and middle-ear cavity have been described. A cross-correlation approach was used to overcome the major obstacle of this processing, the inter-slice misalignment. Although the lack of physical landmarks on the section images makes this strategy prone to error, estimates on the bounds of this error indicate that the resulting alignment is still quite useful. The inclusion of data-reduction techniques into the 3-D model processing path has been shown to be very effective, reducing, for example, the traffic on low-bandwidth channels by at least 50%. The middle-ear model required some sacrifice of accuracy to obtain a polygon model of manageable size. Being an important macro-structure in the vicinity, its inclusion should increase the spatial orientation of the user.

VRML was used to obtain a portable WWW representation for the 3-D anatomical models. The capabilities of current VRML browsers, and their lack of support for high-quality transparent viewing, have been noted. A discussion of the possible ways of implementing transparency led to the selection of the BSP method: it is a highly efficient method of solving the polygon-sort problem, while attaining the realism possible with interpolated transparency. Our implementation of the BSP algorithm, used to produce one of the first

VRML browsers to support "true transparency", was then described and its performance reported. It was observed that the results produced an effective perception of transparency, although it was also noted that diminished colour saturation can accompany transparent rendering. The refresh rates observed were slow using the hardware available to us (0.1-4Hz), but it was anticipated that improvements in low-end hardware would soon enhance both the navigating abilities and the sense of immersion. Possible improvements to the BSP algorithm, such as limiting tree construction during preprocessing, were suggested.

A facility for constructing the 3-D scenes using a WWW form was described. Users can choose the section and structure(s) of interest, and the colour/transparency combinations for each of the surfaces chosen. The inefficiencies of this design were noted: without local management of the scene, any new scene selection requires a complete update from the server. It was mentioned that the capabilities of VRML 2.0 should permit a more efficient design.

8.2 Further work

Although previous work on visualizing ear structures has been carried out, none have done so with the resolution and accessibility achieved by our work. In addition, the easy access and portability now possible with VRML and other WWW tools make this work a reference point for future technological enhancements of the anatomical curriculum. There are many shortcomings, however, that limit the effectiveness of our work as a teaching and learning tool.

Some minor implementation problems need some investigation. As mentioned, the middle-ear cavity model possesses an artifact that can be remedied with further refinement in the segmentation process. The rendering problem, whereby polygons regions are displayed with incorrect colours, likely requires debugging the management of material properties in the BSP code.

From the standpoint of orienting the user using the model, a very helpful feature is the ability to grasp the global context of the data, especially for the very small structures of our

work. Understanding how the slices and structures are oriented in the head could be facilitated by including a head landmark into the 3-D scene, perhaps the skull or the skin. If the difference in scale between the macro and micro structures is considerable, one approach is to have two adjacent views at different scales, with the orientation of the micro structure clearly indicated in the macro scene. Another approach is the use of VRML LOD (level of detail) nodes to vary the degree of model detail to be rendered depending on the position of the viewpoint. From distant viewpoints, the macro scene is rendered completely; the micro scene is rendered very simply, while still giving some indication of orientation. At closer viewpoints, the micro scene is shown in greater detail. This strategy also reduces the waste incurred by detailed rendering of structures that project onto a small image area.

Predefined views that can be quickly selected can also aid orientation. Providing view selections that use terminology such as 'superior', 'inferior', 'medial', 'lateral', 'anterior' and 'posterior' would be most appropriate in the context of these anatomical models. The current default viewing position is a superior view.

Effective labelling of anatomical structures is important for structure identification within the rendered scene. *AsciiText* nodes are part of the VRML specification, allowing 3-D fonts to be positioned in space. As mentioned earlier, these nodes were not supported by *VRweb* at the time of this development, although it has been added to the most recent release (September 1996). This would be the best approach for labeling with anatomical nomenclature. Other associated text (as well the audio and video features of VRML 2.0) could also be included to provide further anatomical concepts related to the region. Addition of the aspect of symbolic concepts to the work could include links to other anatomical scenes having various relationships with the middle ear (regional vasculature, innervation, etc.).

As the amount of information conveyed by an education tool increases, quizzes and drills become more important for learning reinforcement. While the amount of anatomical information in our work probably does not warrant inclusion of such methods, more comprehensive models would not be complete teaching tools without them. Several

methods of student self-appraisal have been described earlier in Chapter 2. They all share the characteristic of motivating the student to evaluate their knowledge and comprehension, and thereby assess their progress.

Our model of the middle ear is certainly far from complete, and is limited to the cat. We have included parts of the tympanic membrane, the malleus, and the middle-ear cavity. The remaining ossicles (incus and stapes), articulating structures, muscles, and the surrounding bone and tissue, to name just a few items, are not present. Detailed display of some of these smaller items is still beyond the limits of current imaging resolutions. Other modalities such as MRI are already giving resolutions comparable to what has been achieved here with histological sectioning. It would be interesting to carry out similar work with these new datasets, and eventually with human data. In order to improve on the inaccuracies we encountered with the marching-cubes method of surface reconstruction, other approaches may be appropriate. Extending the marching-cubes method using multiple transparent surfaces [He *et al.* 1995] merits investigation. Incorporating more reliable and less subjective segmentation methods will also improve the fidelity of reconstructed surfaces to their corresponding raw data.

Data from other modalities also have the benefit of possessing isotropic volumes, so that slicing in arbitrary planes without sacrificing accuracy becomes possible. This ability increases the exploratory possibilities available when portions of the data can be removed to reveal sectioned interior regions: students are closer to the goals of realizing 'virtual dissection'. The dynamic models now possible with VRML 2.0 allow such model transformations to occur.

Models that can change with time bring another benefit: one can display the dynamic nature of a system. The beating of the heart is a common example. Our laboratory has received experimental data measuring the deformation of the tympanic membrane under varying stimulation frequencies, using laser interferometry. It should now be possible to view the membrane moving in response to these inputs, and exploit this type of visualization for

quantitative measurements. The spatial coverage of the measured data is still limited, but should be improving in the near future.

Numerous improvements to rendering quality are also possible. Our work displayed only constant (flat) shading, using a uniform value for the polygon normal. Improved shading is possible when normals are interpolated at the vertices of neighbouring polygons and along the polygon surface itself. Calculation of these normals are often performed by the VRML browser, although *VRweb* did not provide such support at the time of development of our work. For this reason, normal calculation was not incorporated into the BSP rendering development.

Another important efficiency possible with the advent of VRML 2.0 is improved management of scene parameters such as (in our case) slice number, colour and transparency. Only structures that change need be downloaded, although in the present form of our work every new request results in the download of a complete scene. It would be advantageous, for example, to be able to modify transparency values in real time, so that individual structures can be quickly emphasized or de-emphasized. As well, since the slice images can now be defined as separate entities using standard image file formats (rather than the in-line text format used in our work), they can be separately requested and managed. A model-managing script can control such scene updates in memory local to the browser.

Creating useful educational software incorporating these basic suggestions will require considerable implementation effort. Tools available now and those arriving in the future will certainly ease this workload considerably. At a more fundamental level, however, the human role in formulating carefully considered objectives and designs—that ultimately engage and motivate the learning process—will remain a prerequisite for such technological endeavours to be useful additions to the medical curriculum.

References

AAMC, Evaluation of medical information science in medical education. Adopted by the Executive Council of the Association of American Medical Colleges, Washington, DC, January 23, 1986. *Journal of Medical Education*, 61, 1986, p486-543.

Ackerman MF, The Visible Human project of the National Library of Medicine. *MEDINFO92: Proceedings of the Seventh World Congress on Medical Informatics*, KC Lun *et al.*, eds., p366-370. Amsterdam, The Netherlands, North Holland, Elsevier Science Publications, 1992.

Alper EJ and Cardasis C, HistoLogical: a computer atlas and drill of histology, *Proceedings - the Annual Symposium on Computer Applications in Medical Care*, 1991, p935-937.

ANSI (American National Standards Institute), *American National Standard for Information Processing Systems—Programmer's Hierarchical Interactive Graphics System (PHIGS): functional description, archive file format, clear-text encoding of archive file*, ANSI, X3.144-1988, ANSI, New York, 1988.

Appel A, Some techniques for shading machine renderings of solids, *SJCC*, 1968, p37-45.

Apple, *QuickDraw 3D Home Page*, Apple Corporation, 1996, available at `http://quickdraw3d.apple.com/`

Barnett O, Information technology and undergraduate medical education, *Aca Med*, 64(1), 1989, p187-190.

Bell G, Parisi A, Pesce M, The Virtual Reality Modeling Language - Version 1.0 Specification, 1995, available at `http://vrml.wired.com/vrml.tech/vrml10-3.html`

Bouknight WJ and Kelly KC, An algorithm for producing half-tone computer graphics presentations with shadows and movable light sources, *SJCC*, AFIPS Press, Montvale, NJ, 1970, p1-10.

Bouknight WJ, A procedure for generation of three-dimensional half-toned computer graphics presentations, CACM, 13(9), 1970, p527-536.

Callaghan PT and Eccles CD, Sensitivity and resolution in NMR imaging, *Journal of Magnetic Resonance*, 71, 1987, p426-445.

Castleman KR, 1978, *Digital image processing*, Prentice-Hall, Englewood Cliffs, NJ.

Chrysanthou Y and Slater M., Computing dynamic changes to BSP trees, *Computer Graphics Forum (EUROGRAPHICS '92 Proceedings)*, 11(3), 1992, p321-332.

Deubler J, Bastian T, Boulin C, Olivo JC, Automatic registration of serial sections using a wavelet-based multiresolution approach, *Proceedings of the 17th Annual IEEE Conference of Engineering in Medicine and Biology*, 1995, p383-4.

Dev P, Friedman C, Dafoe B, Felciano R, Testing spatial understanding of anatomy, *Proceedings - the Annual Symposium on Computer Applications in Medical Care*, 1992, p 804-805.

Dudgeon DE and Mersereau RM, 1984, *Multidimensional digital signal processing*, Prentice-Hall, Englewood Cliffs, NJ.

Eno K, Sundstem JW, Brinkley JF, A multimedia anatomy browser incorporating a knowledge base and 3-D images, *Proceedings - the Annual Symposium on Computer Applications in Medical Care*, 1992, p727-731.

Foley JD, van Dam A, Feiner SK, Hughes JF, *Computer graphics: principles and practice (2nd ed.)*, Addison-Wesley Publishing Company, USA, 1990.

Fuchs H, Abram GD, Grant ED, Near real-time shaded display of rigid objects, *SIGGRAPH 83*, 1983, p65-72.

Fuchs H, Kedem ZM, Naylor BF, On visible surface generation by a priori tree structures, *SIGGRAPH 80*, 1980, p124-133.

Funnell SM, An approach to finite-element modelling of the middle ear, M.Eng Thesis, Department of Electrical Engineering, McGill University, 1989.

Funnell WRJ and Laszlo CA, A critical review of experimental observations on ear-drum structure and function, *ORL J. Oto-Rhino-Laryngol. Relat. Spec.*, 44, 1982, p181-205.

Funnell WRJ and Laszlo CA, Modelling of the cat eardrum as a thin shell using the finite-element method, *J. Acoust. Soc. Am.*, 63, 1978, p1461-1467.

Funnell WRJ, Decraemer WF, Khanna SM, On the degree of rigidity of the manubrium in a finite-element model of the cat eardrum, *J. Acoust. Soc. Am.*, 91, 1992, 2082-2090.

Funnell WRJ, On the choice of a cost function for the reconstruction of surfaces by triangulation between contours, *Computers and Structures*,18(1), 1984, p23-26.

Funnell WRJ, On the damped frequency response of a finite-element model of the cat eardrum, *J. Acout. Soc. Am.*, 81, 1987, p1851-1859.

Funnell WRJ, On the undamped natural frequencies and mode shapes of a finite-element model of the cat eardrum, *J. Acoust. Soc. Am.*, 73, 1983, p1657-1661.

Goldstein RA and Nagel R, 3-D visual simulation, *Simulation*, 16(1), 1971, p25-31.

Gonzalez RC and Wintz P, *Digital image processing*, Addison-Wesley, Reading, MA, 1977.

Gordon D and Chen S, Front-to-back display of BSP-trees, *IEEE Computer Graphics and Applications*, 11(5), 1991, p79-85.

Gupta SC, Klein SA, Mehl DC Verdi MG, Anderson GL, Introduction of new technologies to the medical undergraduate curriculum, University of Louisville, 1995, available at `http://www.webmed.com/mi/edudoc.html`.

He T, Hong L, Kaufman A, Varshney A, Wang S, Voxel-based object simplification, *Proceedings of the IEEE Visualization Conference 1995*, 1995, p296-303.

Henson MM, Henson OW, Gewalt SL, Wilson JL, Johnson GA, Imaging the cochlea by magnetic resonance microscopy, *Hearing Research*, 75, 1994, p75-80.

Hermans R, Feenstra L, Marchal G, Baert A, Three-dimensional CT-imaging of the ossicular chain, *Clinical Otolaryngology*, 20, 1995, p475-478.

Hibbard LS and Hawkins RA., Objective alignment for three-dimensional reconstruction of digital autoradiograms, *Journal of Neuroscience Methods*, 26, 1988, p55-74.

Hibbard LS, Arnicar-Sulze TL, Dovey-Hartman BJ, Page R, Computed alignment of dissimilar images for three-dimensional reconstructions, *Journal of Neuroscience Methods*, 41, 1992, p133-152.

Higgins G, Hoecht Marion Roussel symposium on virtual reality in the medical curriculum, *Medicine 2001 Conference*, Montréal, Québec, 1996.

Hinker F and Hansen C, Geometric optimization, *Proceedings Visualization 93*, IEEE Computer Society Press, Los Alamitos, CA, 1993, p189-195.

Hoffman H, Irwin A, Ligon R, Murray M, Tohsaku C, Virtual reality-multimedia synthesis: next-generation learning environments for medical education, *Journal of Biocommunication*, 22(3), 1995, p2-7.

Hoffman H, Virtual reality meets medical education, *Interactive Technology and the New Paradigm for Healthcare*, IOS Press and Ohmsha, 1995, p130-136.

Höhne KH and Hanson WA, Interactive 3-D segmentation of MRI and CT volumes using morphological operations, *Journal of Computer Assisted Tomography*, 6(2), 1992, p285-94.

Holdsworth DW, Drangova M, Fenster A, A high resolution XRII-based quantitative volume CT scanner, *Medical Physics*, 20(2), 1993, p449-62.

Intel, *Intel releases new real-time 3D graphics library optimized for the Pentium processor*, Intel Corporation, 1995, available at `http://www.intel.com/pressroom/archive/releases/4-24-95.htm`

Jain AK., *Fundamentals of image processing*, Prentice Hall, Englewood Cliffs, NJ, 1989.

Jennett PA, Edworthy SM, Rosenal TW, Maes WR, Yee N, and Jardine PG. Preparing doctors for tomorrow: information management as a theme in undergraduate medical education, *Medical Education*, 25, 1991, p135-159.

Jih HJ, Reeves TC, Mental models: A research focus for interactive learning systems, *Educational Technology Research & Development*, 40(3), 1992, p39-53.

Johnston R, Hoecht Marion Roussel Symposium on virtual reality in the medical curriculum, *Medicine 2001 Conference*, Montréal, Québec, 1996.

Jonassen DH, Interactive lesson design: a taxonomy, *Educational Technology*, 25(6), 1985, p7-17.

Kaltenborn KF and Rienhoff O, Virtual reality in medicine, *Methods of Information in Medicine*, 32(5), 1993, p407-417.

Kalvin A and Taylor R, Superfaces: polygonal mesh simplification with bounded error, *IEEE Computer Graphics and Applications*, May 1996, 16(4), p64-77.

Lacroute P, The VolPack Volume Rendering Library, 1994, available at `http://www-graphics.stanford.edu/software/volpack/`

Lindberg DAB, Humphreys BL, McCray AT, The Unified Medical Language System, *Methods of Information in Medicine*, 32, 1993, p281-291.

Lorenson WE, Cline HE, Marching cubes: a high resolution 3-D surface reconstruction algorithm, *Computer Graphics*, 1987, 21(3) p 163-169.

Mammen A, Transparency and antialiasing algorithms implemented with the virtual pixel maps technique, *IEEE Computer Graphics and Applications*, 9(4), 1989, p43-55.

Mathematical Applications Group, Inc., 3-D simulated graphics offered by service bureau, *Datamation*, 13(1), 1968, p69.

MathWorks, *Matlab User's Guide*, The MathWorks, Inc., 1992, Natick, MA.

McCormick BH, DeFanti TA, Brown MD, "Visualization in scientific computing", Report of the *NSF Advisory Panel on Graphics, Image Processing and Workstations*. 1987.

Medina R, Garreau M, Jugo D, Castillo C, Toro J, Segmentation of ventricular angiographic images using fuzzy clustering, *Proceedings of the 17th Annual IEEE Conference of Engineering in Medicine and Biology*, 1995, p405-6.

Meinzer HP, cited in: Roux C, Coatrieux JL, Visualization in medicine: virtual reality or actual reality?, *Proceedings Visualization 1994*, IEEE. LosAlamitos, CA, USA, 1994, p396-399.

Merril JR, Notaroberto NF, Laby DM, Rabinowitz AM, Piemme TE, The Ophthalmic Retrobulbar Injection Simulator (ORIS): an application of virtual reality to medical education, *Proceedings—the Annual Symposium on Computer Applications in Medical Care*, 1992, p702-6.

Microsoft, *Microsoft Direct3D backgrounder*, Microsoft Corporation, 1996, available at `http://www.microsoft.com/imedia/direct3d/d3dsheet.htm`

Myers AJ, An efficient visible surface program, *Report to the National Science Foundation*, Computer Graphics Research Group, Ohio State University, Columbus, OH, 1975.

Neider J, Davis T, Woo M, *OpenGL programming guide*, Addison-Wesley, 1993.

Newell ME, Newell RG, Sancha TL, A solution to the hidden-surface problem, *Proceedings of the ACM National Conference 1972*, 1972, p443-450.

Nomura Y, Okuna T, Hara M, Shinagawa Y, Kunii TL, Walking through a human ear, *Acta Otolaryngol*, 107, 1989, p366-370.

Paterson M and Yao F, Efficient binary space partitions for hidden-surface removal, *Solid Modeling, Discrete and Computational Geometry*, (5), 1990, p485-503.

Paul B, The Mesa 3-D graphics library, Space Science and Engineering Center, University of Wisconsin, 1993, Available at `http://www.ssec.wisc.edu/~brianp/Mesa.html` .

Pichler M, Orasche G, Grossman E, McCahill M, *VRweb*: a multi-system VRML viewer, 1995, available at `http://hyperg.iicm.tu-graz.ac.at/VRweb-papers;sk=C6C1311C`

Pradhan M and Dev P, Conceptual change and computer-assisted instruction, *Proceedings of the 17th Annual Symposium on Computer Applications in Medical Care*, 1993, p786-790.

Pratt WK, *Digital image processing (2nd ed.)*, Wiley-Interscience, New York, 1978.

Reed-Ballreich C, Texture mapping in VRML, 3Name3D Inc., 1996, available at
`http://www.ywd.com/cindy/texture.html`

Renze KJ and Oliver JH, Generalized Unstructured Decimation, *IEEE Computer Graphics and Applications*, 16(6), November 1996, p24-32.

Rosenfeld A and Kak A, *Digital picture processing (2nd ed.)*, Academic Press, New York, 1981.

Rosse C, The potential of computerized representations of anatomy in the training of health care providers, *Academic Medicine*, 70(6), 1995, p499-505.

Schroeder W, Martin K, Lorensen W, The Visualization Toolkit (vtk), 1996, available at
`http://www.cs.rpi.edu/~martink`

Schroeder W, Martin K., Lorensen W., *The Visualization Toolkit*, 1996, Prentice-Hall, USA.

Schroeder W, Zarge J, Lorensen W, Decimation of triangle meshes, *Computer Graphics*, (Proc. Siggraph), 26(2), July 1992, p65-70.

Schubert R, Höhne KH, Pommert A, Riemer M, Schiemann T, Tiede U, Lierse W, A new method for practicing exploration, dissection, and simulation with a complete three-dimensional model of the brain and skull, *Acta Anatomica*, 150(1), 1994, p69-74.

Schumacker R, Brand B, Gilliland M, Sharp W, Study for applying computer-generated images to visual simulation, *Technical Report AFHRL-TR-69-14*, NTIS AD700375, U.S. Air Force Human Resources Lab, Air Force Systems Command, Brooks AFB, TX, 1969.

Schwartz EL, Merker B, Wolfson E, Shaw A, Applications of computer graphics and image processing to 2-D and 3-D modeling of the functional architecture of visual cortex., *IEEE Computer Graphics and Applications*, 8(4), July 1988, p13-23.

SDSC (San Diego Supercomputing Center), The VRML Repository, 1996, available at
`http://www.sdsc.edu/vrml/repository.html`

Shortliffe E and Perreault L, *Medical informatics: computer applications in health care*, 1990, Addison Wesley, USA.

Sprague LA, Bell B, Sullivan T, Voss M, Payer AF, Goza SM, Virtual reality in medical education and assessment, *Proceedings of the fifth workshop on neural networks: academic/industrial/NASA/defense—an international conference on computational intelligence: neural networks, fuzzy systems, evolutionary programming and virtual reality*, 1993, p373-377.

Stanford W; Erkonen WE, Casssell MD, Moran BD, Easley G, Carris RL, Albanese MA., Evaluation of a computer-based program for teaching cardiac anatomy, *Investigative Radiology*, 29(2), Feb. 1994, p 248-252.

Strauss PS, QVLib: a VRML parser, Silicon Graphics, 1995, available at
`http://vrml.wired.com/vrml.tech/qv.html`

Tiede U, Schiemann T, Höhne KH, Visualizing the Visible Human, *IEEE Computer Graphics and Applications*, 15(1), 1996, p7-9.

Toejeck J, Department of Otolaryngology, University of Wisconsin, 1996, available at
`http://www.biostat.wisc.edu/otoweb/otoweb.html`

Torres E, Optimization of the binary space partition algorithm (bsp) for the visualization of dynamic scenes, *Eurographics '90*, Elsevier Science Publishers (North Holland).

Vinitski S, Gonzalez C, Burnett C, Buchheit W, Mohamed FB, Ortega HV, Faro S, 3-D segmentation in mri of brain tumors: preliminary results, *Proceedings of the 17th Annual IEEE Conference of Engineering in Medicine and Biology*, 1995, p481-2.

Wade B, BSP Frequently asked questions, 1994, Qualia Incorporated, available at
`http://www.qualia.com/bspfaq/`.

Warnock J, A hidden-surface algorithm for computer generated half-tone pictures, *Technical Report TR 4-15, NTIS AD-753 671*, Computer Science Department, University of Utah, Salt Lake City, UT, 1969.

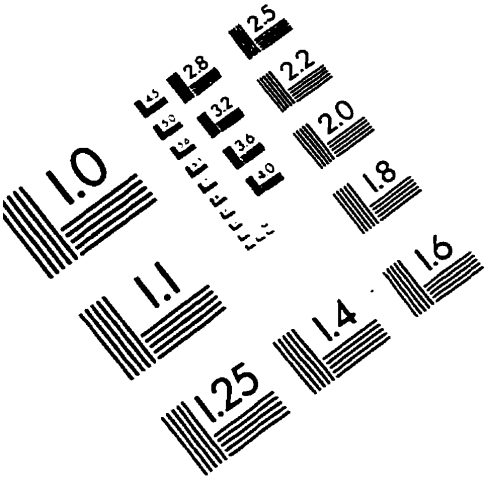
Warscotte V, Macq B, Thiran JP, Michel C, Accurate segmentation of 3-D Magnetic Resonance Images of the head using a Directional Watershed Transform, *Proceedings of the 17th Annual IEEE Conference of Engineering in Medicine and Biology*, 1995, p491-2.

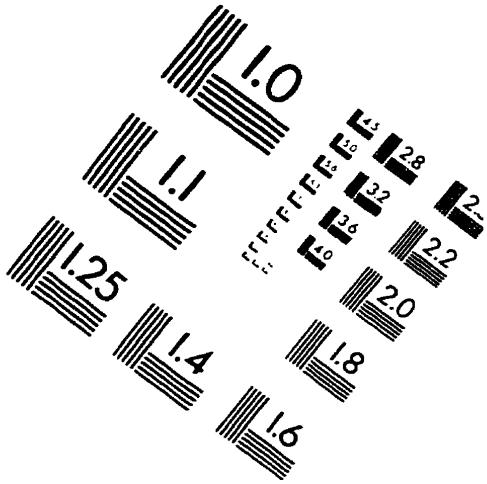
Watkins GS, *A Real-time Visible Surface Algorithm*, Ph. D. Thesis, Technical Report UTEC-CSc-70-101, NTIS AD-762 004, Computer Science Department, University of Utah, Salt Lake City, UT, 1970.

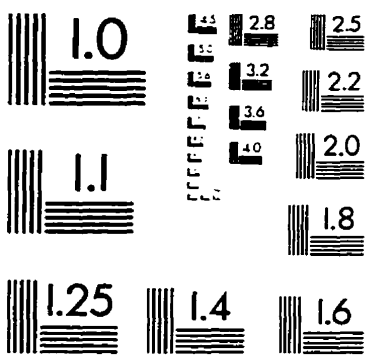
Weiler K, Atherton P, Hidden surface removal using polygon area sorting, *SIGGRAPH 77*, 1977.

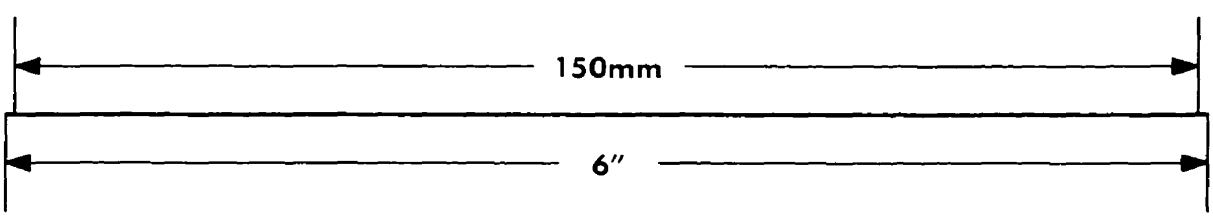
Wylie C, Romney GW, Evans DC, Erdahl AC, Halftone perpective drawings by computer, *FJCC 67*, Thompson Books, Washington, DC, 1967, p49-58.

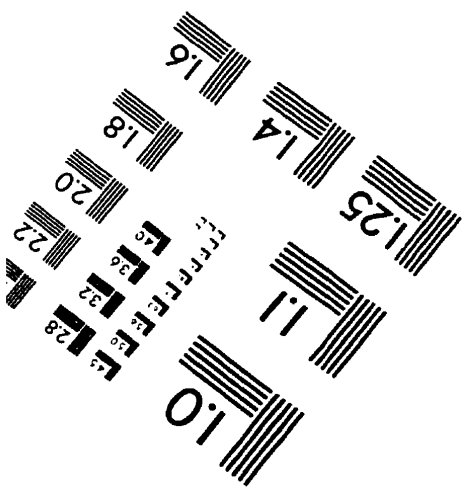
IMAGE EVALUATION
TEST TARGET (QA-3)

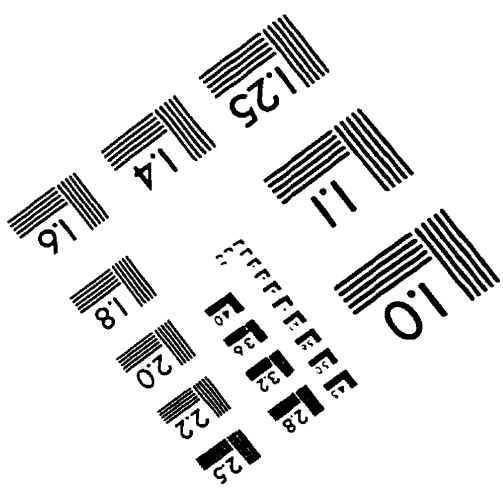












APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989