Integrated medical-image system for cancer research and treatment

by R. Ohbuchi
T. Miyazawa
M. Aono
A. Koide
M. Kimura
R. Yoshida

This paper describes a large, heterogeneous network of supercomputers, workstations, and personal computers for clinicians and researchers at the National Cancer Center in Tokyo. Intended uses of the system include a medical-image database, a patient-record database, visualization of 3D medical images, computer-assisted diagnosis, genome and protein databases, and proof-of-concept synthetic environments. An overview of the system is followed by detailed descriptions of two subsystems: the medical-image-database subsystem and the synthetic-environment subsystem. The medical-image-database subsystem integrates a large number of image-acquisition devices, such as X-ray computed tomography and ultrasound echography equipment. The image database is designed for a high image-data input rate by means of a storage hierarchy and high-speed network connections. The synthetic-environment subsystem includes two prototype applications (a neurosurgery simulator and an anatomy viewer) that explore the possibilities of synthetic-environment technology in medicine.

1. Introduction

Computers have been playing an ever-increasing role in every aspect of our lives, and medicine is not an exception. Computers have been used in medicine to manage databases of patient records, to reconstruct images for X-ray computed tomography (CT) systems, to manage databases of medical images taken by X-ray radiography and other imaging modalities, as well as for other purposes[1].

A system recently acquired by the National Cancer Center (NCC) of Tokyo, Japan, a nationally chartered hospital dedicated to cancer treatment and research, is a heterogeneous, networked system of supercomputers, workstations, and personal computers that performs a multitude of tasks for clinical and research communities located both inside and outside the hospital. The system consists of many subsystems, including a medical-image-database subsystem, a computer-assisted-diagnostics subsystem, genome and protein database subsystems, and a synthetic-environment subsystem.

A proposal for an integrated, networked medical-information system prepared in 1993 by the clinicians and researchers at NCC with the help of a consulting firm had the following four areas of emphasis:

[1] See also the paper by Taylor et al. in this issue.

0018-8646/96/$5.00 © 1996 IBM

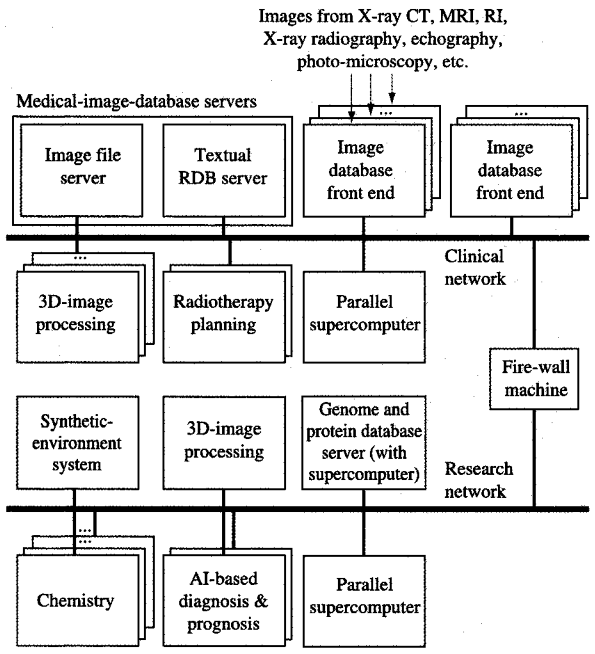

Figure 1

Overall configuration of the system installed at the National Cancer Center of Tokyo, Japan.

1. *Medical-image systems*:
   a. Image-database subsystem.
   b. 3D-image analysis and diagnosis subsystem.
   c. Radiotherapy-planning subsystem.
   d. AI-based diagnosis and prognosis subsystem.
   e. Synthetic-environment subsystem.
2. *Clinical-information systems*:
   a. Clinical database subsystem.
   b. Statistical-analysis subsystem.
   c. AI applications.
3. *Biomedical systems*:
   a. Genome and protein database subsystems.
   b. Computational-chemistry subsystem.
   c. Molecular analysis and design subsystems.
4. *Cancer-information-provider system*:
   Information servers to disseminate cancer information to researchers, clinicians, and the general public through the network.

IBM Japan made a bid based on this proposal and was awarded a contract for a system-integration project by NCC in February 1994.

This paper focuses on the medical-image systems, in particular, the image-database subsystem (1a) and the synthetic-environment subsystem (1e), in which the authors are directly involved. The other three areas (clinical information systems, biomedical systems, and cancer-information-provider system) are not discussed further in this paper. Most of the work in those areas consisted of the integration of available software and hardware systems.

To give an overall perspective, the following subsection presents the hardware configuration of the heterogeneous network of computers that is used by all four application areas. The next subsection briefly describes various subsystems included in the first area, the medical image system. Section 2 presents the details of the medical-image-database subsystem, and Section 3 discusses details of the synthetic-environment subsystem. The paper ends with a summary.

• *Hardware configuration of the entire NCC system*

The hardware system installed in the NCC is a heterogeneous network of three supercomputers, 112 workstations, and 85 personal computers (PCs) (**Figure 1**). The most prominent among the hardware components are the following three supercomputers:

- *IBM 9076 (SP2™)*[2] *parallel computer*, with 40 nodes—medical-image processing and molecular design.
- *IBM 9076 (SP2) parallel computer*, with four nodes—statistical analysis and medical-image processing.
- *DECmpp™*[3] *12000/sx Model 200 parallel computer*, with 16384 processors—genome and protein databases.

In addition, each of the two Silicon Graphics Inc. (SGI) Onyx graphics workstations has four processors, giving each workstation near-supercomputer performance. Other workstations are various models from IBM, Digital Equipment Corp., SGI, and Sun Microsystems. The 85 PCs are all models of the Apple Macintosh™.

In order to handle a very large data volume, the database system has a very large total hard disk capacity. For example, the image file server has 100 GB (gigabytes) of hard disk capacity. The hard disks and a rewritable magneto-optical (MO) disk "jukebox" with a 900GB capacity, attached to the image file server, form a storage hierarchy. The MO disk jukebox (CYGNET 1803 from Advanced Archival Products, Inc.) is managed by AMASS™ (Archival Management and Storage System) software. In addition, two 8-mm tape jukeboxes (EpochBackup™ system from Epoch Systems, Inc.), whose capacity with compression is approximately 1 TB (terabyte) per jukebox, are used to back up the database.

[2] Scalable POWERparallel™ System 2, now called RS/6000™ SP.
[3] Digital Equipment Corp. massively parallel processor.

All of these supercomputers, workstations, and PCs are interconnected by one or more communication links. (The storage systems are connected to the computers—not directly to the links.) The following networking protocols are used: Ethernet, Fiber-Distributed Data Interface (FDDI), and High-Performance Parallel Interface (HIPPI). For example, one of the SP2s and two of the Onyxes are connected by all three of the interconnection protocols. Concerns over the privacy of patients caused the network to be segmented into two sections, the *clinical network* and the *research network*, which are bridged by a firewall machine, a computer designed to carefully control the communication between the two networks in order to maintain system security. The clinical network is the secure segment, whose access is limited. The NCC plans to make the research network available to outside research and clinical institutions, so that the databases and the computational resources can be shared. In addition, the NCC plans to have a server on the research network disseminate cancer information for laypersons.

◆ *Components of the medical-image system*

The medical-image system is used for day-to-day clinical work by doctors, nurses, and researchers. It is arguably the most significant component of the entire NCC system. The following list gives an overview of the five subsystems of the medical-image system:

- The *image-database subsystem* is the largest and the most important element of our medical-image system. The intended uses of the database include diagnosis, treatment planning, treatment-efficacy assessment, research, and teaching. Image data are acquired from a number of imaging devices, which acquire images by various means, e.g., X-ray radiography, X-ray CT, and ultrasound echography. Databases are accessed by doctors, nurses, and technicians through database-access workstations located in offices, laboratories, and operating rooms. At a workstation, a patient record from the clinical database can be displayed on a screen side by side with the images and the data associated with them (called *image attributes*), retrieved from the image database. The details of the image-database subsystem are described in the following section.
- The *3D-image-analysis and diagnosis subsystem* is intended to help researchers and clinicians in diagnosis and research through advanced 3D visualization. This subsystem consists of workstations and several visualization software packages[4]. The image-database subsystem has a number of format-conversion routines to accommodate the many different image formats used by the visualization software packages.

  We have created parallel versions of some of the modules of the AVS package in order to take advantage of the supercomputing power of the SP2s for image analysis and 3D visualization. Parallelized modules include an iso-value contour-surface-generation module, a direct-volume-rendering module, and various image-processing modules[5]. The parallelized modules run as remote modules on one of the SP2s, while the users interact with AVS running on local workstations.
- The *radiotherapy-planning subsystem* is used to plan cancer treatment by irradiation from either outside or inside the body. The subsystem currently consists of two commercial turnkey systems[6]. Both of the systems use SGI workstations as their platforms and are largely self-contained, including digitizer tablets, printers, flatbed scanners, and other peripherals. These systems are connected to the medical-image database, so that the images in the database can be used for treatment planning.
- The *AI-based diagnosis and prognosis subsystem*, an experimental system intended for research, explores the applicability of neural-network analysis and rule-based reasoning to prognostication of cancer treatment. The system has been implemented by using commercial toolkits for neural-network and rule-based-reasoning systems.
- The *synthetic-environment subsystem*, also experimental, is intended to explore the applicability of synthetic-environment technology to medicine through prototypes of two immersive virtual-environment systems: a neurosurgery-simulator system and an anatomy-viewer system. The details of this subsystem are described in Section 3.

The medical-image system uses a significant portion of the total hardware resources of the NCC project, including both of the SP2s, 72 workstations, and 21 PCs. Of these, 25 of the workstations and the SP2s, which must transfer large amounts of image data, are interconnected by FDDI cables via a high-speed switch. FDDI has roughly ten times the throughput of the older Ethernet.

Of the five subsystems, the image-database, 3D-image analysis and diagnosis, and radiotherapy-planning subsystems handle patients' clinical information. Consequently, workstations and PCs for these three subsystems are connected to the secure clinical network behind the firewall. Workstations and PCs used for the remaining two subsystems handle nonclassified data only and are connected to the research network.

[4] AVS, from Advanced Visualization Systems, Inc.; Dr. View, from Asahi Kasei Joho Systems Co., Ltd.; Volume Design Pro, from Medical Design Inc.; Voxel View, from Vital Image, Inc.; VIPstation, from CEMAX Inc.; and CliPSS, from the IBM Tokyo Research Laboratory [1].

[5] Readers interested in visualization techniques for volume data by surface generation and by direct volume rendering are referred to references on graphics, such as [2]. Two books [3, 4] discuss the subject of scientific visualization.

[6] PLATO, from Nucletron International B.V., and FOCUS, from Computerized Medical Systems Inc.

Subsequent sections of this paper present details of two of the subsystems, the image-database subsystem and the synthetic-environment subsystem.

2. Image-database subsystem

The image-database subsystem of the NCC project provides a system infrastructure for the hospital to manage medical images and their image-attributes, both for research and clinical purposes. The image database allows the doctors and their staffs to enter, retrieve, view, and annotate images and related records for diagnosis, treatment, and research purposes. The database is accessed from workstations located near the users in offices, laboratories, and operating rooms. The images and related records come from a variety of imaging equipment, located throughout the NCC, that acquires images in many different ways (modalities).

The image database employs a number of workstations and PCs used as the database front ends, in addition to a pair of centralized database servers. The database-access front-end application program, called the Clinical View Station (CVS), runs on the workstations. The CVS handles acquisition, entry, caching, transfer, and display of data. The centralized database servers manage the database; they store, find, retrieve, update, and back up the data.

Few, if any, medical-image databases rival in their complexity and scale the NCC medical-image database, which posed several technical challenges to us.

One of the challenges is the need to manage a large volume of medical image data. For example, it takes 10 MB (megabytes) without compression to store a set of magnetic resonance imaging (MRI) images consisting of 20 slices of 512×512-pixel images at 2 bytes per pixel. The same images, if compressed using a lossless JPEG (Joint Photographic Experts Group) algorithm, are about 75% of the size of the original images. (Doctors normally require lossless compression, in order not to degrade the quality of images used for diagnosis and treatment.) A survey of workers at NCC prior to the system design revealed that every day the image database system is expected to store up to 4 GB of additional image data. Fast processors, fast networks, large main memory, large disk space, and a storage hierarchy are the necessary hardware components to handle this requirement. Details of the database servers are described in later subsections.

Another of the challenges is to interface directly with a multitude of imaging modalities, which include X-ray CT, MRI, various means of radiological imaging (RI) such as single-photon-emission computed tomography (SPECT), ultrasound echography, endoscopy, as well as various kinds of microscopy equipment used in the pathology department. This imaging equipment uses many different formats for images and their related records, but the data must be converted to have a common image-data format for transfer and storage in the database. We have defined a common file format for the project, called the NCC format, which is an extension of the Digital Imaging and Communications in Medicine (DICOM) 3.0 file format [5].

The DICOM 3.0 format is a standard format for the storage and transfer of digitally encoded medical images. It was developed by the American College of Radiologists (ACR) and the National Electrical Manufacturers Association (NEMA) as a successor to ACR/NEMA 1.0/2.0. For our purposes, the DICOM 3.0 format displays several shortcomings. It does not define the handling of multiple images, such as a set of images that, as a group, represents a volume. It also does not clearly define the handling of compressed image data. Using a method similar to the Papyrus file format [6], developed at the University Hospital of Geneva to handle groupings of multiple images, we have defined rules to handle these cases within the framework of the DICOM 3.0 format. We also added a set of rules so that JPEG-compressed images are stored in the pixel data field of the DICOM 3.0 format. Finally, we added the ability to transcribe names in Japanese characters. (We hope that in the future all standards likely to become international will include national-language support.)

The image database is accessed from the CVS program running on database-access workstations located in offices, laboratories, and operating rooms. The medical-image database consists of two physical servers, a *textual relational-database (RDB) server* and an *image-file server* (see Figure 1). A typical query by a doctor would first search for a patient by name in the textual RDB. This query would retrieve one or more textual records of studies done for the patient. The attributes in a study record include the name and birth date of the patient, the patient number, the name of the resident and attending doctors, and pointers to the images generated by the study. When images are to be viewed, a study of interest is selected, and the image files for the study are retrieved from the image database and transmitted to the database-access workstations, to be viewed by means of the CVS. Users can also access a separate *clinical-information database* for the patient's clinical record, which contains information not necessarily related to images, such as drugs administered in the past. The information in the clinical record and the images can be viewed side by side on a workstation monitor. The details of the CVS are discussed below.

The database-access workstation permits several modes of operation, in addition to the typical entry, search, and retrieval operations on the image database. First, since a clinical conference may call for a very rapid viewing of many images, the user may prefetch and store several images in the database-access workstation. Second, the images may be directly transferred among database-access workstations by means of the file transfer protocol (FTP), without going through the database. Third, the retrieved images may be exported with format conversions to other visualization software such as CliPSS and AVS.

The first subsection that follows presents the steps needed for acquisition of image data from imaging devices and image entry into the database. The second subsection describes the image-database server itself, and the third discusses the CVS.

• *Data acquisition and entry*

The term *data acquisition* refers to the process of acquiring images and the data associated with them (image attributes) from imaging equipment. If the imaging equipment does not supply enough information regarding the image, it must be obtained from other sources, e.g., from the users. There are three classes of data acquisition devices, organized according to how image data are passed from the imaging devices to the database system: via network file transfer, via video signal, or via film.

Devices in the first class export image data and their attributes, as computer-readable files in various formats, through network connections to workstations that serve as gateways to the database. This class of devices includes X-ray CT, MRI, and RI scanners. Network connectivity is still not common among these devices, so special interface boxes are often necessary to provide the connection. For each study, the physicians or technicians enter image attributes from the console of the imaging equipment, for example, the patient's ID, the part of the body being imaged, the scanning order (e.g., from head to toe), and a short comment. Scanned image data and their attributes are temporarily stored in the imaging equipment and later transferred to the database-access workstation connected to the device. This transfer is initiated manually several times a day.

Devices in the second class export image data as NTSC video signals. This type of device includes ultrasound echography equipment, endoscopy equipment, and video cameras attached to microscopes in the pathology department. Video signals are digitized by video digitizer boards installed in the PCs connected to the imaging devices. Digitized images are automatically written on the disks of the database-access workstations through the network.

Devices in the third class export images in film form. Devices in this class include those for conventional X-ray radiography. In this case, film scanners attached to some of the database-access workstations are used to digitize films.

The following steps that occur on the database-access workstations are the same for every class of device. As the images and their image attributes arrive at the database-access workstation, their formats are converted to the NCC format. Upon reformatting, images and their image attributes that belong to a *study* are grouped into an entity, and *study attributes* that describe the study are created and added to this entity. A study in this case is, roughly speaking, a set of images (e.g., X-ray radiograph pictures) and their attributes (e.g., patient name, patient number, aspects of the photograph) for a patient, with a specific purpose (e.g., to diagnose a lump in the neck), usually on one occasion.

Converting into the NCC format may leave image attributes unspecified, depending on the devices. The devices in the first class have only a few empty slots. The devices in the second and third classes, on the other hand, have almost all of the image attributes empty. The missing image-attribute items are determined by accessing patient records in the clinical-information database. If patient records do not supply necessary data, the users must enter the missing attributes.

After the attributes are supplied, doctors and operators view the images in order to select those images that are important enough to be stored in the medical-image database. To avoid mistakes, image attributes of the images picked for storage are then validated by consulting the clinical information database. The CVS program performs the validations automatically by making sure the name, birth date, and patient number match between the two databases. Those images selected to be stored in the database are compressed automatically on the database-access workstation, by means of either reversible or irreversible encoding options of the JPEG standard.

The transfer of images selected for storage from the database-access workstations into the image database servers occurs automatically, typically at midnight, when the loads of the network and the database servers are the lightest. Prior to their registration into the database, images in the database-access workstations cannot be referenced by the other workstations. If the images are needed immediately, however, a user can force the transfer and registration of images into the image-database server[7].

The image database assists doctors in some of their practices. For example, prior to the introduction of the image database, doctors marked a few crucial images, called *key frames*, in each study so that they could be pulled from files quickly. (For example, an endoscopic study typically acquires a total of 20 to 60 images, not all of which are considered important.) The image database supports this practice with the key-frame-marker attribute. The image database also supports the approval process

[7] The pathology department is unusual in that it has its own local database server; image data and their image attributes are stored in the local database before being passed on to the (central) image database. This configuration is based on the requirements of the doctors in the pathology department.

Table 1 Estimation of data volume produced. Images from the RI and pathology departments have multiple sizes and formats.

Department	*Studies per day*	*Images per day*	*MB per day*	*Notes*
CT	39	2040	1020	2D or 3D data
MRI	8	440	55	2D or 3D data
RI	34	110	53	Multiple types of scanners, 2D or 3D data
Film	130	130	860	High resolution
Echography	60	1310	400	Monochrome and color
Endoscopy	50	2000	300	Color
Pathology	—	140	70	Color
Total	321	6170	2758	

("signing off") by the responsible ("attending") doctors of the diagnoses made by the doctors in training ("residents").

• *Image-database server*

As discussed in the previous subsection, the image database consists of images and their attributes. The image database can be searched by composite queries formed as combinations of attributes. Options exist for defining ranges (for example, of dates) and using "wild cards." The database, however, does not have the capability to search by image content.

As mentioned before, the image database server physically consists of two servers: the textual RDB server and the image-file server. The textual RDB, developed using the Oracle database system [7] from Oracle, Inc., holds all of the image attributes and study attributes. Oracle runs on an IBM RS/6000™ computer with 256 MB of memory and 50 GB of hard disk. Images themselves, which are pointed to by the entries in the textual RDB, are stored in the separate image-file server, which consists of an IBM RS/6000 computer with 512 MB of main memory, 100 GB of hard disk, and the 900GB MO disk jukebox. The clinical-information database, which keeps patient records, runs on another IBM RS/6000 computer with 256 MB of memory and 50 GB of hard disk.

The separation of the RDB server and the image-file server has yielded a reasonable load balance between the retrieval of image attributes and the retrieval of the images. The separation also has made it easier for the images to be managed in a storage hierarchy. While it is not a part of the image database, the clinical-information-database server works in concert with the image-database servers. The clinical information database, however, is used primarily for the AI-based diagnosis and prognosis systems.

As mentioned before, a significant characteristic of this database is the large volume of image data stored every day. **Table 1** shows the estimated daily data volume, based on a user survey, for each of the seven departments of the NCC. As we have developed the system, we have made performance measurements of the RDB server by loading the database with artificial data for 1.3 million images, the estimated number of images added each year. The tuning of the database was based on these measurements.

In order to manage the large volume of image data, we adopted hierarchical data management in three layers; the images are stored in the hierarchy, either on the hard disks, in the MO-disk jukebox, or in the 8-mm-tape jukeboxes. For newly arrived images to be stored promptly, the image-database server must maintain enough empty space in each level of the hierarchy. In this database system, the least-recently-used files are migrated from the hard disks to the MO-disk jukebox. The same policy is used to purge files from the MO-disk jukebox, after they are written onto 8-mm-tape jukeboxes. If users want to access files already purged from the disks and the MO-disk jukebox, the files must be restored from the archives in the 8-mm jukeboxes. When these jukeboxes become full, some tapes may be removed and shelved, later to be reloaded as needed.

To the projected data rate of 2.8 GB per day (from Table 1), we added a 40% safety margin, so that the data rate assumed for the design was 4 GB per day. Lossless JPEG compression was expected to reduce this number to 1.5 GB per day. At a data rate of 1.5 GB per day, the hard disk drives would become full after about 12 weeks (if initially empty), and the MO jukebox would become full after about two years. The user survey showed that, most of the time, data for a patient are accessed for somewhere between one week and one month. We thus concluded that the estimated lifetime of an image file on the hard disk, about 12 weeks, is sufficiently long to allow quick access to the images for most patients. It should be noted that key frames constitute about one fifth of the images, and have expected lifetimes of more than ten years before they are purged from the MO jukebox.

• *Database-access front end*

CVS is the set of programs that provides a graphical user interface for the image-database system. Users can query the database, view and manipulate retrieved images and their image attributes, or export retrieved images to software packages for 3D visualization. CVS runs on the database-access workstations, which are located in offices, laboratories, and operating rooms. The functions of CVS are the following:

- Perform RDB operations, such as selection, on study and image attributes.
- Edit attributes.
- Retrieve and annotate images. An annotation consists of a set of graphic primitives, measurements, and marks to be overlaid on a displayed image.
- Display and edit images.
- Manage retrieved images and attributes on the local disk, archiving and deleting as appropriate.
- Transfer image and attribute files directly to or from other workstations by use of ftp.
- Convert the format of images in order to exchange data with software packages.

The users of the system are non-computer-oriented professionals, that is, doctors, nurses, and laboratory technicians. We chose a menu-based user interface for the CVS in order to minimize typing[8]. In implementing the user interface, flexibility is crucial, since numerous variations in the dialog box, panels, and such are called for. These variations in the user interface stem from such factors as seven user-access privilege classes (for example, one group of doctors may write diagnoses, while the others may only view them), departmental differences in procedures, and varieties of imaging equipment involved.

In addition to ease of use, we tried to offer powerful functions in the CVS for the viewing, annotation, and manipulation of images. The following list summarizes the main functions of CVS:

- Accommodating four pixel data types (1-bit black-and-white, 8-bit gray, 16-bit gray, and 24-bit RGB color).
- Automatically detecting display adapter hardware modes and adjusting image presentations by "dithering" colors.
- Rapid, arbitrary-scale zooming of images.
- Real-time, continuous change of the center and width of the "contrast window." (In CVS, a contrast window specifies a piecewise-linear function that maps the pixel values of the original data onto display pixel values. Using this function, doctors can display more clearly slight variations in contrast that are barely noticeable without such an adjustment.)

[8] Emphasis on reduction of typing may be stronger in Japan than, for example, in the U.S.A. Typing is much less common in Japan than in most countries with smaller alphabets (26 characters or so). Keyboard input was popularized only about ten years ago, with the advent of personal computers and word processors that could refer to electronic dictionaries of words and grammars to convert typed syllables into text.

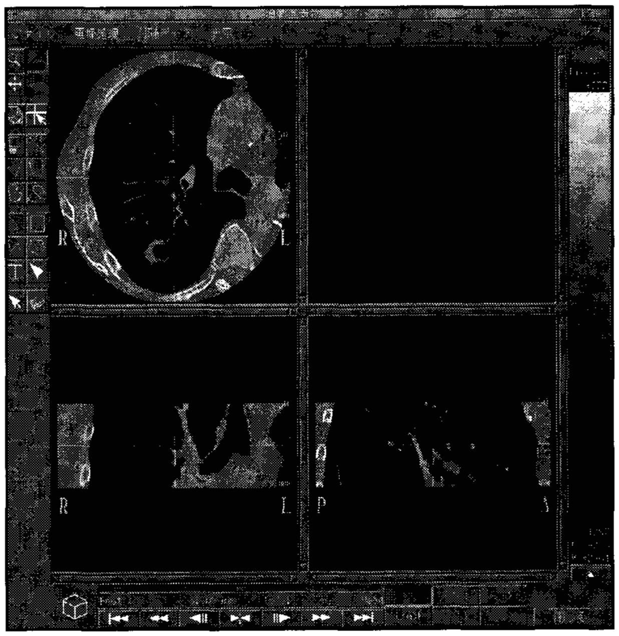

Figure 2

Transverse (top-left), coronal (bottom-left), and sagittal (bottom-right) slices of a volume image of the lung, acquired by an X-ray CT scanner, are displayed by the Clinical View Station.

- Displaying volume images by slicing through the data in three planes (sagittal, coronal, and transverse planes, in anatomy).
- Displaying multiple-slice images with movie-loops (the rapid display of a sequence of slice images, like an animation).
- Annotating images with five graphic primitives (line segment, polyline, polygon, rectangle, and text), in a manner similar to object-based drawing programs such as Claris MacDraw™.
- Measuring pixel values, distances, angles, areas, and lengths of traced contours with six built-in tools.

The CVS program must run on different makes of workstations with several different kinds of display-system configurations. For this reason, the programming effort paid special attention to portability. For example, we used the X Windows® System and Motif® for display management.

Figure 2 is an example of a CVS window displaying a volume image, acquired by X-ray CT, by slicing the volume by three planes. The images on display are computed locally on the workstation, based on the volume image data retrieved from the image database. The three large windows at the center of the panel display the sliced images (the upper-right-hand subwindow is blank in this viewing mode). A vertical strip to the right shows the zooming ratio, the color-lookup table (gray scale in this case) with the minimum and maximum values of the data, and the center and the width of the contrast window. Using buttons on the left side panel of the CVS window, a user can select several graphic functions for annotation and several measurement functions. Using buttons and a slider on the bottom control panel, the user can select which slices to display.

• *Summary of image-database system*
We developed the image-database system as a part of the NCC system-integration project. The system provides the infrastructure to manage a large volume of medical images and their attributes for diagnosis, treatment, and research purposes. Images come from medical-imaging equipment of various kinds. The doctors and their staffs may enter, retrieve, view, and annotate the digital images and their attributes on the local workstations in offices, laboratories, and operating rooms.

In order to manage this large volume of images, which increases at an estimated rate of 2.8 GB per day, we use hierarchical data management in three layers, which consists of hard disks, an MO-disk jukebox, and an 8-mm-tape jukebox. A fast network, i.e., FDDI, and powerful database servers are other crucial components in realizing the image database.

The user interface of the CVS is menu-based, so that accessing the database is easy for non-computer-oriented professionals. Powerful functions, combined with the user-friendly interface available in the CVS, allow the users to edit and annotate images with ease.

• *Discussion of the image-database system*
Many issues of a technical nature, both significant and insignificant, arose after the system went into use. We discuss a few of the issues that we found interesting.

Requirements for security and usability are often at odds. In the NCC system, privacy and security concerns caused the network to be segmented into two sections separated by the firewall: the clinical network (secure segment) and the research network (less secure segment). This segmentation caused usability problems. For example, the research network includes the larger of the two SP2s (40 nodes), the DECmpp, and the two Onyxes, to make them available to the users inside and outside the NCC. However, this makes it very awkward, if not impossible, to use these resources from the secure clinical network behind the firewall. We have yet to find a solution to this problem.

Another security vs. usability conflict we experienced was the reluctance of many of the users to accept a password-based security system. A security measure for the medical-image system requires all of the users to remember and use their own user identifications (IDs) and passwords. If a user fails more than a few times to provide the correct password, the workstation is locked, and a system administrator is notified through a beeper.

An additional explanation for user resistance to the user-ID/password system is the redundancy perceived by the users in the security measures. In addition to the medical-image system, a pre-existing computer system called *Hospital Information System (HIS)* has been used to deal with patients' accounting information, appointments, and other applications. This system provided security through physical exclusion of unintended users (access limited because of fixed and known locations of terminals) combined with personal ID cards with magnetic stripes. From the user's viewpoint, having two separate methods to maintain security must have been perceived as cumbersome. One of the users complained, "Why can't I use the ID card (for the HIS) that I carry around anyway?"

It could be dangerous to rely solely on ID cards for identification because of the danger posed by misplaced or forged cards. However, in the future, it will probably be advisable to consider and present a coherent security control method to the users. Such unification may not be easy, since the HIS is a proprietary system owned by another company.

As with many other projects, we could not correctly predict changes in user practice due to the installation of the new system. After the system went into operation, we discovered that actual daily data volume has exceeded the projection we used during the system design. Instead of the 2.8 GB per day we used for the design, the actual value has risen (and stabilized so far) to about 4 GB per day. Fortunately, thanks to the safety factors existing in the system, we have dealt with this issue by simply increasing the frequency of archiving onto the 8-mm tape. The increase in terms of the burden on the operator due to the increased archiving has been insignificant.

• *Lessons learned from the image-database system*
As we have discussed previously, data acquisition from a multitude of imaging devices was a challenge. Most of this equipment was not designed with networking and data-sharing in mind. Many of the devices could not export their images in computer-readable form via networks. Even if they could export data in computer-readable formats, the file formats they output were proprietary and incompatible with one another.

To deal with multiple image formats, we developed image-format conversion programs that run on PCs and workstations. This is where we spent an unexpected amount of effort; we had problems writing the conversion programs because of the inexact and imprecise specifications of the proprietary file formats.

While we followed written specifications of file formats received from the manufacturers of the imaging equipment, when we started testing the format conversion programs we experienced numerous errors for reasons such as illegal syntax and out-of-range values. Imprecise documentation, undocumented features, undocumented updates, and other documentation problems were the cause. Some of the errors were discovered only after weeks of testing with esoteric combinations of image-acquisition parameters. Other errors cropped up unexpectedly after software upgrades.

We cannot overstate the importance of knowing the exact, precise file formats. Perhaps we should have established closer ties with the manufacturers in order to obtain detailed and updated documentation of file formats.

A better alternative for the future would be to establish an international standard format to be adopted by all manufacturers of medical-imaging equipment. This may take some time, however. For example, one of the more recent proposed standard formats, the DICOM 3.0 format, does not have built-in provisions to handle national languages such as Thai and Japanese.

3. Synthetic-environment subsystem

• *Background*

The goal of the synthetic-environment (SE) subsystem in the NCC project was to explore the possibilities of SE technologies applied to medicine. We chose to develop a proof-of-concept neurosurgery-simulator system for the NCC SE project. We also developed an SE analog of an anatomy book, called the *cancer-information theater*. The remainder of this paper concentrates on the neurosurgery-simulator system. Discussion of the cancer-information theater is limited to a short description in the subsection on the synthetic-environment system.

For the last ten years or so, researchers have been investigating the use of computer and 3D medical images in planning, optimizing, and rehearsing various surgical procedures, for example, craniofacial surgery. Pioneering systems of this kind have required a long time simply to visualize the patient anatomy or the result of simulated operations. With the advent of graphics engines with increased performance, such as the Silicon Graphics RealityEngine [8], it has now become possible to visualize relatively complex medical images in 3D at reasonable speeds.

Even with such powerful graphics workstations, however, most of the previous surgical planning or simulation systems could not perform simulated surgical operations in real time, nor view the effect of the operation in real time. Users of such systems must often wait a significant time until the simulated surgical operation is performed on the models of anatomy stored in the computer and the results of the operation are visualized. While there are a few surgical simulation systems that run at an interactive rate, they use generic models of anatomy, so they are useful for training only. When actual patient data is used for surgical simulation at or near an interactive update rate, the quality of the visualization is often compromised (for example, [9]).

The long-term goal of the project on the neurosurgery simulator described in this paper is to realize a system to plan, simulate, optimize, and rehearse neurosurgical operations. As a first step toward this long-term goal, we decided to develop a proof-of-concept neurosurgery simulator with three key features. The first feature is the capability to visualize the anatomy of the brain with high quality by using images of patients captured in 3D by medical imaging modalities such as MRI. The second feature is the capability to simulate neurosurgery operations. The third feature is the ability to perform the visualization and the simulated surgical operations at an interactive rate in an immersive virtual environment. Note that, in this paper, *interactive rate* means a simulation and visualization update rate of more than ten frames per second, which is generally regarded as the minimum rate needed to give users the feel of immersion in a 3D world.

In the following sections of this paper, we first define such terms as *synthetic-environment, virtual-environment*, and *immersive* systems, for the term *virtual reality* is ill-defined and often abused. We then review previous work on synthetic-environment technology applied to medicine. Our neurosurgery simulator system is described next, starting with a typical simulation scenario and followed by the description of hardware and software systems, with an emphasis on the *minimal toolkit for virtual environment*, which is the toolkit we have developed for the construction of virtual-environment applications. The algorithms we used to implement various operations necessary in the neurosurgery simulator are described next, including high-quality visualization techniques for the brain and a method to efficiently approximate cutting and other simulated surgical operations. We conclude the paper with a summary and discussion.

Taxonomy

While the term *virtual reality* (*VR*) has been used frequently in the media, it is not well defined. The authors prefer the term *synthetic environment* as defined in [10] as an inclusive term. SE systems, which can be classified into the three subcategories listed below, are human–machine interface systems that exploit multimodal information input and output capabilities as well as cognitive capabilities of human beings.

Table 2 Various types of virtual-environment (VE) and augmented-reality (AR) visual displays are compared with one another and with unaided human vision (HV), on the basis of the visual information provided. In the table, "visual" data are those visible to the unaided human eye, while "sensory" data are acquired with the aid of machines. Both vision and sensory data arrive in real time, while stored or generated data are either stored in, or generated by, the computer.

System types	*Data sources*			*Application examples*
	Visual	*Sensory*	*Stored or generated*	
VE	No	No	Yes	Architectural walk-through, molecular modeling
VE	No	Yes	No	Scanning tunneling microscope visualization [12]
VE	No	Yes	Yes	?
HV	Yes	No	No	Human vision
AR	Yes	No	Yes	Laser printer repair [13], aircraft wiring harness assembly [14]
AR	Yes	Yes	No	Medical ultrasound imaging [11]
AR	Yes	Yes	Yes	?

- A *tele-operator* (*TO*) system uses an SE interface so that one or more human beings can sense and affect real-world objects that are otherwise not directly manipulable. For example, a TO system can control an underwater robot for bridge construction or an atomic-force microscope to realign molecules.
- A *virtual-environment* (*VE*) system uses an SE interface to present synthetic information that is generated by and/or stored in a computer. An example of a VE system is an architectural walk-through of a building yet to be built. The large majority of existing VR systems fall into this category.
- An *augmented-reality* (*AR*) system combines information from the real world with information from a synthetic world in real time. For example, to guide surgery, computer-generated images of a brain tumor can be displayed "in place"—that is, at the proper location within the patient's head on the operating table, while the surgeon, wearing a head-mounted display, moves around the patient. The synthetic information to be merged with real-world information can be either stored, static information (for example, schematics of a jet engine), or sensory, dynamic information, requiring real-time processing (for example, echographic images of a fetus [11]). **Table 2** compares visual displays of AR and VE systems.

For example, the neurosurgery simulator described in this paper is a VE system.

The advantages of SE interfaces over conventional human–computer interfaces derive from the increased number and quality of interaction modes employed. In an SE system, sensory inputs to human beings may consist of one or more visual, auditory, tactile, kinesthetic, and other stimulations. Outputs from human beings may employ methods that are identical to those used in everyday life, e.g., direct manipulation by hands and other body parts, speech, a sign language based on gesture, and point-and-click action on a 3D menu.

So far, however, the bulk of the work on SE systems has been focusing on the use of enhanced visual-display techniques that exploit capabilities of the human visual system, most noticeably by using head-mounted displays (HMDs) [15] or "fish-tank VR displays" [16, 17]. Both HMDs and fish-tank VR displays can provide visual 3D cues, by means of binocular disparity, convergence, and motion parallax, in addition to the monocular 3D cues (occlusion, linear perspective, texture gradient, shading, and others) provided by conventional desktop video monitors. Other visual 3D cues, such as ocular accommodation, can also be important but are difficult to provide by using visual display devices currently available.

In order to provide binocular disparity, a typical HMD device employs two image-display elements (e.g., two liquid-crystal display panels) with proper optics, mounted on the user's head. In order to provide motion parallax, the HMD system requires real-time tracking of the locations and orientations of the user's head. A fish-tank VR display typically consists of a desktop time-multiplexed stereo video display monitor combined with viewpoint tracking. The displays also require high-throughput, low-latency image generation (or acquisition, if it is used in a TO system) based on the tracking information.

These displays for SE can be classified as either *immersive* or *non-immersive*. An immersive display completely encloses the user with the images presented in the display device. An HMD without see-through capability is an example of an immersive display. A fish-tank VR display is a non-immersive display. A see-through HMD, in which the user can see the real world and virtual world overlaid in some way (for example, optically [15] or electronically [11], is another example of a non-immersive display. Interaction techniques are different among systems that use immersive and non-immersive displays. With a non-immersive display, a user can interact with objects seen in the real world, such as a 2D window menu or a (2D) mouse on a real desk top. However, with the immersive display, interactions must all take place in the 3D world in which the user is immersed.

The number of studies dedicated to nonvisual display techniques is much smaller than those on visual display techniques. Work has been reported on "auditory display" (e.g., a spatially localized, or "3D," sound), inertial display (e.g., motion platforms of flight simulators), force display (e.g., a joystick that kicks back, or an "active joystick"), and proprioceptive/kinesthetic displays (e.g., a treadmill).

It should be noted that the technologies for SE are still immature. Some even claim that the field has not advanced much since the first seminal work by Sutherland [15]. For example, current HMD systems often fail to provide convincing 3D cues by head-motion parallax because of such factors as system lag, position and orientation tracking errors, lack of scene complexity, and poor image quality. While the authors believe that SE technologies at the present stage of development can be put to good use, the application programs must be carefully designed to take advantage of the still-limited technology. Readers interested in SE-interface technologies are referred to [10, 18, 19], books that provide good starting points.

Synthetic environments and medicine

There are numerous examples of computer systems put to use in medicine. A large number of them involve 3D imaging. Some even combine 3D imaging, 3D graphics, computer-aided design, and robotics in order to plan and robotically execute hip-joint-replacement surgery [20]. The display techniques employed by most of those systems, however, are not SE techniques.

Only a small number of researchers have employed an SE interface with an immersive, head-tracked display with motion parallax. Examples of such SE interfaces in medicine include diagnostics [11, 21], simulation, planning and training for various kinds of surgical procedures [22–26], psychotherapy [27, 28], and improving the quality of life of hospitalized patients (for example, through entertainment [29]).

In terms of computer assistance (not limited to systems with an SE interface) for surgical procedures, neurosurgery has probably been the most popular field [30–36]. Examples range from surgical planning with the use of a traditional desk-top monitor, and mechanical intra-operative guidance of surgical instruments to the location of interest (e.g., tumors), to the AR-like fusion of real images (i.e., video image of patients) and virtual images (e.g., volume-rendered images of tumors).

One of the most interesting examples is an AR-like guidance system for brain tumor removal surgery that has been put to clinical use [32, 35]. This example is categorized as only "AR-like," since the viewpoint is fixed in order to remove most of the difficulties of "real" AR systems, such as tracking error, time-lag, and finding the shapes and positions of real objects.

To the authors' knowledge, however, there is not yet a system for neurosurgery simulation (or guidance) which actually employs head-tracked, immersive VE (or AR) that is based on models generated from data acquired by 3D medical imaging equipment such as MRI scanners.

• *Our synthetic-environment system*

The focus of our neurosurgery simulation system is the exploration of the capabilities of an immersive VE interface (with a dynamic viewpoint). Among the neurosurgical operations, we picked craniotomy (open-skull surgery) to remove a tumor relatively close to the skull surface.

In overly simplified terms, such a craniotomy typically involves the following steps: 1) cutting the skin to expose the skull, 2) drilling holes and sawing between the holes to remove part of the skull, 3) cutting membranes to expose the brain tissue, 4) cutting the brain tissue with a suction tube and pushing aside the brain tissue with spatulas so that the tumor is exposed, 5) aspirating the contents of the tumor (if necessary), and resecting the tumor. To facilitate the patient's recovery, the damage to the skin, skull, and brain tissue should be minimal. Vital structures, such as major arteries, veins, and crucial functional regions of the brain, must not be damaged.

The long-term goal of the neurosurgery simulator is a system that helps doctors plan, optimize, rehearse, and learn neurosurgery techniques through visualization of, and simulation on, the brain anatomies of individual patients. Our current proof-of-concept surgical simulation system is only the first step toward such a system. Our neurosurgery simulator visualizes, in a VE, a simplified model of the human head, which includes models of the skin, skull, brain, and a tumor. An important feature of the system is that these models are generated from 3D medical-image data. In the past, 3D MRI images of the brain have been displayed in 3D with non-SE display interfaces. However, we think that our system may be the first to display an image of the brain created from 3D MRI data, with reasonable quality, by means of a head-tracked (dynamic viewpoint) VE interface. In addition to the visualization, our system allows the users to plan crude approximations of craniotomy operations for removing a tumor and then to practice them by controlling virtual surgical instruments to cut the skin, drill holes in the skull, and remove the brain tissue.

Note that in the proof-of-concept neurosurgery simulator reported in this paper, the model of the anatomy is greatly simplified so that the current generation of graphics-rendering engines can create the

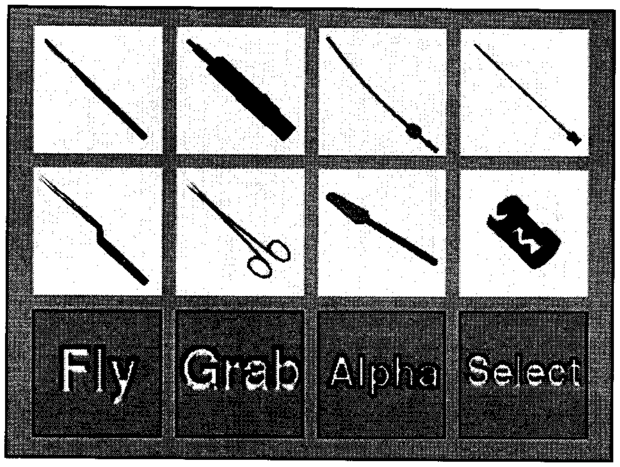

Figure 3

This menu pops up at the location of the tracker (a stylus) when the button of the tracker is double-clicked. The user selects an operation from the menu by clicking on one of the icons.

visualization at an interactive update rate. Similarly, the simulation of surgical operations is also simplified so that they fall within the capability of current computer systems.

Another VE application we have built, the cancer-information theater, is intended to educate patients (to facilitate the process of obtaining informed consent) by providing an interactive learning tool for various forms of cancer. The system is essentially a VE version of an anatomy book for a layperson, with the emphasis on cancer. Wearing an HMD or a pair of LCD-shutter stereo glasses, a patient might select from an immersive 3D menu an organ of interest. The system would then respond by delivering information, for example, by presenting a 3D model of the organ together with photomicroscopic pictures of the cancerous tissues of the organ. Simultaneously, a voice message would explain the risk factors of, the diagnostic methods for, and the treatment modalities for the cancer. Further description of the cancer-information-theater system is not given in this paper, since it uses a program identical to that of the neurosurgery simulator. The differences between the two systems lie in their world descriptions, which specify the objects and their behavior by using the *virtual-environment file (VEF)* format we have defined.

A typical surgical simulation scenario

The first step of the simulated craniotomy (to remove a small brain tumor near the brain surface) is to determine the location of the tumor, which can be accomplished by making the skin, the skull, and the brain semitransparent. After planning the surgery, the doctors would carry out a simulated surgery to rehearse and evaluate the plan.

The patient's body is modeled only above the neck, in order to reduce the number of polygons that must be processed. (A model of a sheet covers the nonexistent part of the body below the neck.) The head consists of four independent objects: the skin, the skull, the brain, and a tumor. Other important objects in the head, such as the eyes, optic nerves, major blood vessels, and dura mater (a membrane underneath the skull), are not included in the current model. These and other simplifications in shapes and behaviors of the objects in this virtual world were necessary in order to reduce the cost of modeling, simulation, and rendering. For example, we decided at an early stage to exclude the model of the body for performance reasons. Another reason for the simplification was that some of the omitted objects, for example, the blood vessels, are difficult to generate automatically from 3D medical images. Besides the head, the virtual world for this surgical simulation scenario includes models of the operating theater and the operating table, which are necessary to give the users spatial references and a familiar environment.

The system allows users to invoke various actions by pointing and clicking on 3D menus, as well as by direct manipulation (e.g., grabbing and cutting). A double click of a button on a six-degree-of-freedom (6-DOF) tracking device toggles on and off the 3D-pop-up main menu shown in **Figure 3** (actually a 2D menu in a 3D world). A single click while a menu item is highlighted launches the operation associated with the menu item. For example, selecting the "Alpha" icon in the main menu invokes a submenu to adjust transparencies of objects. **Figure 4** shows the skin, skull, and brain, along with the submenu for adjusting the transparency in order to find the position of the tumor. The arrow in front of the menu is a 3D cursor, whose shape changes to that of a scalpel and other instruments depending on the context.

According to the type of surgery planned, after locating the tumor in the head, the user may grab and reposition the head appropriately (e.g., to a face-down position), as in **Figure 5**. In a real operation, a jig is used to immobilize the head during the surgery; however, such a jig is not included in the current simulation scenario.

Please note that we used test data (from a hypothetical patient) for the models of the virtual world that are shown in this paper; the model of the brain is generated from MRI data, while others are artificial. (The MRI data for the brain were obtained by courtesy of Siemens A.G. and the University of North Carolina at Chapel Hill.)

After positioning the head, the user selects a scalpel icon, which turns the 3D cursor into a scalpel. A single click of the button activates the scalpel so that it can cut the skin. The active status is indicated to the user by the

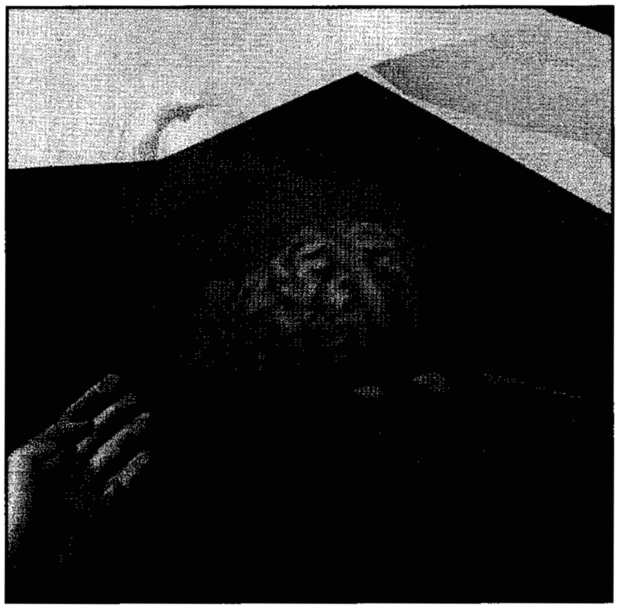

Figure 4

The skin, skull, and brain were made semitransparent for locating the tumor. The menu adjusts the degrees of transparency. The arrow is the 3D cursor, which changes its shape depending on the context.

Figure 5

The user grabs and rotates the head so that the envisioned area of entry is exposed for easy access. During real surgery, the head is immobilized by a jig.

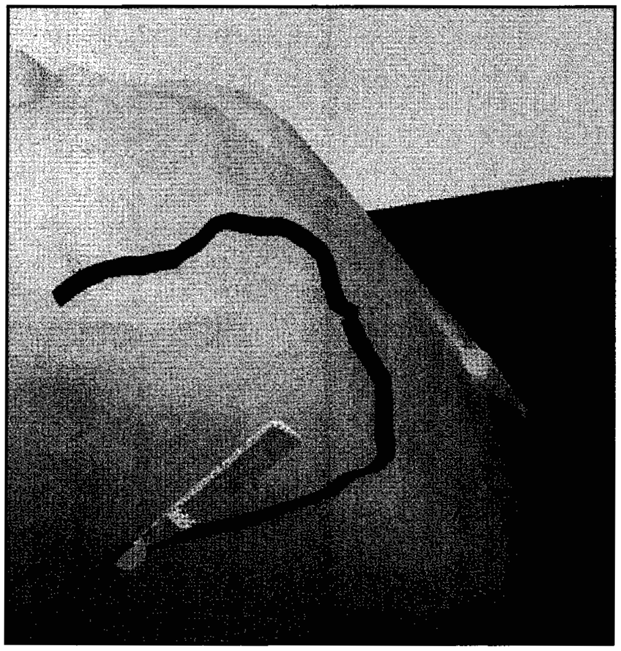

Figure 6

The operator drags the virtual scalpel over the skin, marking the line to be cut as a poly-line. After the marking, the user clicks a button on the scalpel to actually cut out the area specified.

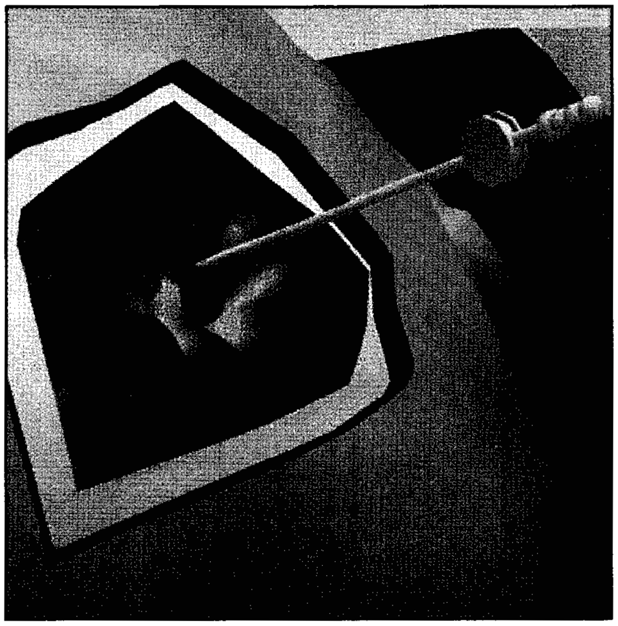

Figure 7

The skin and the skull-bone are cut open, and the virtual suction device evacuates brain matter (visualized from MRI data) to expose the tumor (an artificial model) for resection.

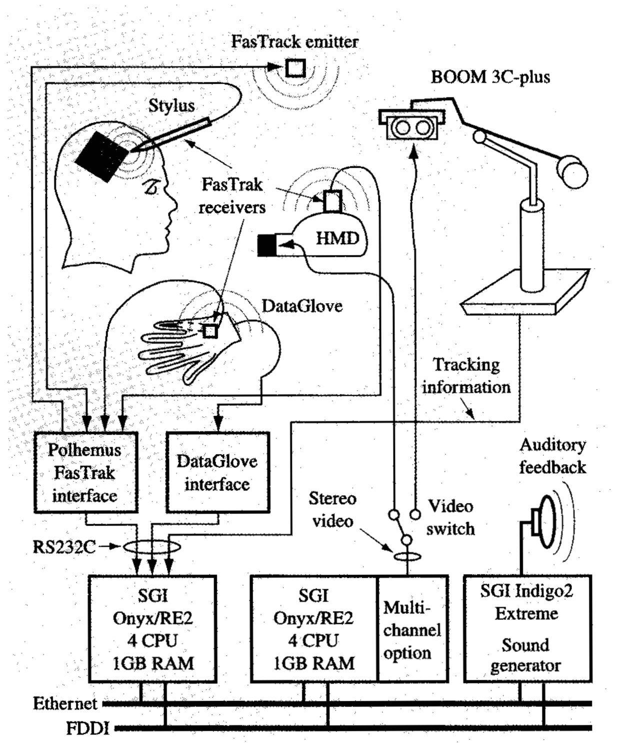

Figure 8

Hardware configuration of the virtual-environment system.

scalpel's turning green. The cutting of the skin is very much simplified in this simulation, as are drilling and other surgical procedures. As the user drags the activated virtual scalpel across the skin along the edges of the intended opening, the path of the scalpel is indicated by a black line (see **Figure 6**). It is necessary (but not sufficient) to provide the user with instantaneous audible feedback by means of the sound and the drawing of a black incision line, since there is no force or tactile feedback to indicate contact of the scalpel with the skin. After the line has been marked, a click of the button instantaneously removes the skin enclosed by the black line.

Next, the user chooses the drill from the menu in order to open the skull. The process of opening the skull is approximated by marking the vertices of the intended hole on the skull with the virtual air-drill bit, followed by a single clicking of the button. As feedback, a drilling sound is generated as the drill bit touches the skull.

The next step is to open a path through the brain tissue to reach the buried tumor. In actual surgery, a suction tube repeatedly "cuts" the brain tissue by evacuating brain tissue, and spatulas are used to push aside the soft tissue of the brain to expose the tumor. In the simulation, the path is opened simply by excavation with the suction device (**Figure 7**). Models of neither the blood vessels nor the deformation of the brain tissue are included in the simulation. As the user excavates, or "digs," the brain tissue, folds of the gray and white matter internal to the brain become visible because of the visualization method that employed solid texture [37, 38].

After the tumor is exposed, it may be aspirated by the aspiration needle in order to shrink it before resection. In the simulation, the aspiration is simulated simply by decreasing the size of the tumor, and the complex process of resection is not simulated at all. After the tumor is reduced, it is picked up and removed from the brain.

Hardware

Figure 8 shows the hardware configuration for our VE system. We use a pair of SGI Onyx workstations, each with four CPUs, 1 GB of memory, 12 GB of disk space, and a graphics accelerator, which is a single-pipe RealityEngine 2 (RE2). One of the Onyx/RE2s has the Multi-Channel Option (MCO), so that more than two images from different viewpoints can be generated, for example to drive two video displays for binocular-stereo viewing. The MCO produces multiple video outputs from different areas of a physical frame buffer.

Display devices include a FakeSpace Laboratory Binocular Omni-Orientational Monitor (BOOM) Model 3C-Plus and a Nissho EyePhone HRX Model J2 (an HMD). Three console monitors and a projector are equipped with Stereo Graphics CrystalEyes LCD-shutter glass stereo display devices to be used as high-resolution stereo displays without head-tracking.

The locations and orientations of the HMD are tracked by a receiver of a Polhemus FasTrak 6-DOF magnetic-tracker device (three translational and three rotational degrees of freedom), while the locations and orientations of the BOOM are tracked, with 6-DOF, through its mechanical tracking arm by using its built-in optical encoders. Other FasTrak receivers are attached to a stylus with a button and two gloves made of elastic fabric. The stylus may control a 3D cursor or one of the virtual surgical instruments, depending on the context of the simulation. Single- or double-clicking of the button on the stylus selects or executes items from 3D pop-up menus, activates virtual instruments, or executes actions associated with the menus or virtual instruments. The glove, developed by Nissho, is able to measure the angles of ten finger joints for each hand. A separate workstation provides real-time auditory feedback, such as a drilling sound, to the system.

Software

The neurosurgery simulator program was written in C with the help of a VE-development toolkit [called minimal toolkit for virtual environment (MTVE)] we developed for the project. This section presents the overview of the software system. Some of the features of the MTVE toolkit are described in the following sections.

Figure 9 shows important functional modules of the neurosurgery simulator and their relationships. The application program (enclosed in the large rectangle in Figure 9) is executed as a collection of processes. The processes communicate via shared memory, so that the application program could exploit four shared-memory processors in each of the Onyx workstations. Input- and output-related processes may be distributed over one or more workstations. Inputs from devices such as the FasTrak tracker are handed to the application program from their respective server processes via communication paths provided by the Virtual Reality Distributed Environment and Construction Kit (VR-DECK) [39]. Requests for auditory rendering from the application program are also sent through the VR-DECK to the sound server running on its separate workstation, an Indigo2 Extreme.

The VR-DECK is an experimental toolkit for distributed programming, with typed, message-based communication. With VR-DECK, processes written in C (or C++™) can communicate via a global name-space over a group of workstations connected by a network. In addition to providing a communication infrastructure, the VR-DECK includes several device servers for input devices (e.g., FasTrak) for VE systems. The VBUILD graphical user interface of the VR-DECK provides an easy way for a user to do plumber's work on the communication pipes. However, VR-DECK is not a typical VE toolkit (e.g., [40–42]), for it does not support modeling, object database management, rendering, or display management. All of these missing functions must be supplied by the user code; in our case, they are provided by the MTVE toolkit and the SGI IRIS Performer visual simulation library [43].

A problem we faced with the VR-DECK is its first-in-first-out (FIFO) message ordering. Typically, a VE system needs only the most recent data from trackers, so that the lag from the trackers to image generation can be minimal. We circumvented the FIFO ordering by using a shared memory in the tracker-interface process of the application program.

The MTVE toolkit

The MTVE is a program library for VE development that we have created by using IRIS Performer. The MTVE toolkit manages loading of, modification of, and interaction with the virtual environment. Each virtual world is described by an editable text file that uses the virtual environment file (VEF) format we have defined.

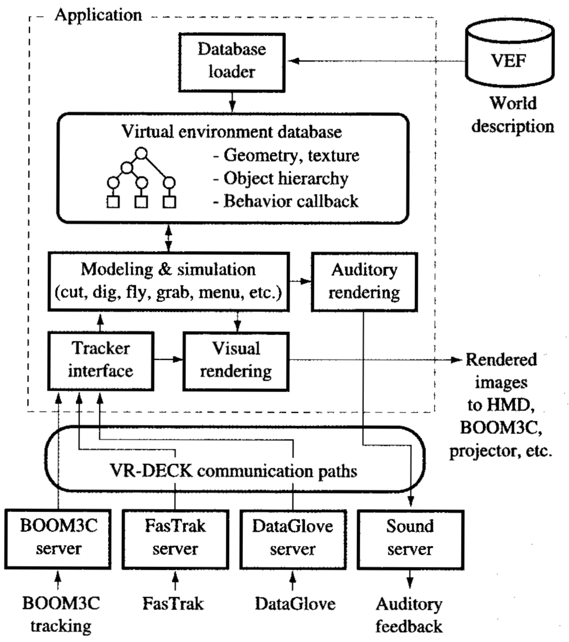

Figure 9

Structures of functional modules in the surgical simulation system.

A VEF defines a virtual world—that is, all objects and their relationships in the virtual world, and the interactions between the objects and the user. The VEF defines, among other characteristics, the geometries, positions, colors, textures, and behaviors of objects, and their hierarchical relationships. A behavior of an object is specified by writing a callback routine and attaching it to the object. The callback routine is associated with the name in the VEF via a user-supplied table. The MTVE has several built-in behaviors that are common in a virtual environment, such as "flying," "grabbing," and "changing transparency." It supports solid texture for rendering objects to allow high-speed and high-quality visualization of the brain from medical 3D MRI data. The MTVE also supports simple hierarchical menus, in which 2D icons in the 3D space can be selected by pointing and clicking with the 3D cursor. In addition, unlike all of the other VE toolkits available at the time, the MTVE toolkit provides functions necessary for neurosurgery simulation, e.g., cutting the skin by using the scalpel and evacuating the brain tissue by using the suction tube.

World description with VEF

Figure 10 shows an excerpt of a VEF that defines the `head` object, which consists of the `face`, `skull`, and

```
# Definition of the head.
# The head consists of brain, face, and skull objects.
#
OBJECT brain {                        # A node that groups its children.
   GROUP {}
   POSITION {
      ROTATE     ( 0.0, 0.0, 0.0 )
      SCALE      ( 1.0, 1.0, 1.0 )
      TRANSLATE  ( 0.0, 0.0, 0.0 )
   }
}
OBJECT brainG {
   MODEL {
      GEOMETRY   "MRbrain.dcel"      # TDCEL geometry file from the MRI data.
      TEXTURE    "MRbrain.stex"      # Solid texture from the MRI data.
      MAPMODE     solid              # Map the texture as a solid texture.
      AMBIENT    ( 0.3, 0.3, 0.3 )
      DIFFUSE    ( 0.7, 0.7, 0.7 )
      SPECULAR   ( 0.8, 0.8, 0.8 )
      SHININESS  8.0
   }
   POSITION {
      ROTATE     ( 90.0, 0.0, 0.0 )
      SCALE      ( 2.0, 2.0, 2.0 )
      TRANSLATE  ( 0.0, 0.0, 0.0 )
   }
   BEHAVIOR {
      COLLIDE            on                         # Collidee objects
      COLLIDE_MASK       4                          # with mask 4.
      OPERATION          dig                        # Can be excavated
      COLLIDE_CALLBACK   "mtveDigObjectCallback"   # by calling back this routine.
      COLLIDE_DATA_SOUND "sound_dig_brain"          # Generate this sound upon collision.
   }
}
OBJECT skullG{
...
}
OBJECT faceG{
...
}

# Hierarchy definition
STRUCTURE head { # The head consists of the brain, the skull, and the face.
   CHILD   brain
   CHILD   skull
   CHILD   face
}
STRUCTURE brain {
   CHILD   brainG
}
STRUCTURE skull {
   CHILD   skullG
}
STRUCTURE face {
   CHILD   faceG
}
```

Figure 10

An excerpt from a virtual environment file that shows an example definition of the head. (Definitions of *skullG* and *faceG* are omitted for simplicity.)

brain objects. A VEF consists of definitions of objects with various attributes (the OBJECT part) and definitions of the nodes' hierarchical relations (the STRUCTURE part). A VEF may include other VEFs. For example, a definition of an object (e.g., a hand) can be replicated by writing one VEF for the object and including a reference to the file whenever it is called for in another VEF.

There are visible nodes and invisible nodes. Visible nodes have geometrical shapes and other visible attributes, such as colors and textures. Invisible nodes include light sources, pairs of user's eyes, and group nodes that group their children. Sets of nodes form hierarchical relationships in a virtual world. These relationships are also hierarchies of modeling transformation, the root of the tree being the world coordinate origin. (Please refer to [44] for the basics of computer graphics.) In the example of Figure 10, if the head is rotated, face, skull, and brain rotate along with it.

Attributes of a visible object node (e.g., skull) include its geometric shape, color, texture, texture-mapping method, initial position and orientation, and scale. The MTVE provides conventional planar texture as well as solid texture for visualization of the brain from 3D MRI data (see the subsection on visualizing the brain). A light source has position, direction, color, and intensity, but not geometry. A node may be associated with a 6-DOF tracker, to follow the tracker's movement; for example, an eye node has as its attributes the name of the tracker to which it is attached. The eye also has the interocular distance for binocular stereo image generation.

Detection of and reaction to collisions among objects are very important in implementing a reactive VE. Such collision detection is necessary in cutting the skin, for example. The MTVE utilizes a simple but effective collision-detection program built into IRIS Performer. In the MTVE, each node with geometrical shape (called a "collidee") may collide with invisible line segments (called "intersection segments") that are attached to nodes (called "colliders"). For example, to simulate the drilling operation, the skull object may collide with an intersection segment attached to the tip of the air drill. A collider may not intersect with every collidee; for example, the scalpel collides with the skin, but not with the skull. Whether a collidee may intersect with an intersection segment is controlled by the COLLIDE_MASK attribute of the collidee.

The action to be taken upon collision is specified by a callback function name. The callback function is typically provided by the programmer, but the MTVE has built-in actions, such as *fly*, *grab*, *cut*, *dig*, *scale*, and *change-transparency* as well as simple 3D pop-up menu handling, that can be invoked through collision detection. Algorithms for some of the actions are described in the following section. A sound to be emitted upon collision may be specified for an object, to provide immediate feedback to the users (e.g., the "sound_dig_brain" for the brainG object in Figure 10).

Another important functionality supported by the MTVE is support for hierarchical 3D pop-up menus. As mentioned before, an immersive VE requires all of the interaction to take place in the VE. The MTVE allows the user to describe hierarchical, 3D pop-up menus in the VEF as a part of the virtual world description. A menu is a kind of VEF node with predefined geometry. The menu pops up at the location of the 6-DOF tracker corresponding to the 3D cursor, always facing the user. Figure 5 shows an example of a menu, which is an array of 2D icons. The VEF description of the menu for each icon includes the texture, the icon's spatial arrangement, operation name, and name of the callback function for the action.

The MTVE toolkit with the VEF makes development of new VE applications quite easy. This was demonstrated through the development of the cancer-information theater, which took only three days. While the cancer-information theater and the surgical simulator are significantly different in appearance and in function, they rely on the MTVE toolkit and differ only in their VEF descriptions. Their "development" consisted entirely of preparing the data objects and writing and tuning the VEF description.

• *Visualization and simulation methods*

Visualizing the brain

To be useful for planning and rehearsing surgery of actual patients, the neurosurgery simulation system must visualize 3D medical-image data of patients, for example data obtained by MRI. The visualization method should provide sufficient anatomical detail for doctors to understand the location of the tumor and plan the path to access the tumor for removal. It must be fast enough to provide an immersive virtual environment when implemented on the available hardware. Simulation of surgical operations must also take place at an interactive speed.

We have considered many combinations of simulation models and rendering models. Simulation models can be classified as either surface-based or voxel-based models. Examples of surface-based models are collections of polygons and free-form curved surfaces. Voxel-based models include simple enumeration by voxels and hierarchically encoded forms by using such data structure as octree [45].

Voxel models seem natural for MRI and X-ray CT data, which are by themselves voxel-based 3D image data. Voxel models permit easy implementation of set operations (e.g., union and intersection), although their display quality tends to be low and their data size large. Voxel models

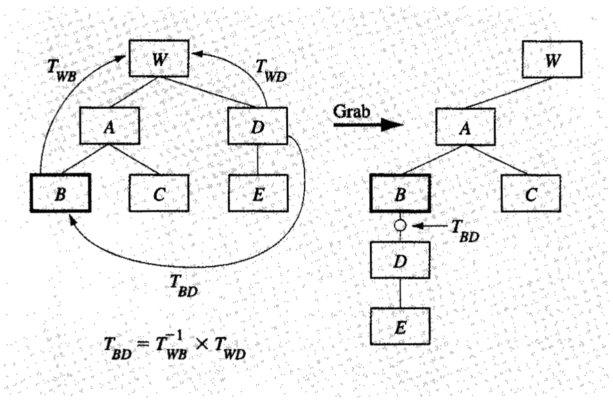

Figure 11

Object hierarchy before (left) and after (right) object B grabs object D.

can be rendered at interactive speed by either 1) direct volume rendering [46, 47] or 2) converting them into surface models [48]. If the data are modified dynamically, however, neither of these methods will be able to yield interactive performance for simulation and rendering.

Another approach is to convert the voxel-based 3D images from the medical-imaging equipment into polygonal-surface models (for example, by the marching-cubes algorithm [49]) and use the latter for simulation and rendering. One drawback of this method is that if details of the brain shape, such as the configuration of the folds, are modeled by geometry, the number of polygons for the model becomes prohibitively high for interactive simulation and rendering.

We have adopted a polygonal-surface model with a small number of polygons, whose visual appearance is enhanced by a solid texture. By defining 3D MRI data as the solid texture, we can add significant detail to the visualization of the brain surface, which consists of a relatively small number of polygons. A solid texture is also able to show the interior of the brain when simulated tissue removal is performed by displacing brain-surface polygons. (See the subsection on cutting and digging.)

Each of the polygonal surface models that is either cut or dug is paired with a TDCEL (*triangular doubly connected edge list*) data structure to add edge and vertex-adjacency information. TDCEL is a doubly connected edge list (DCEL) [50] specialized for triangles. The adjacency information makes efficient the simulated cutting and digging operations. For example, given a surface model that consists of N triangles, without adjacency information, finding a triangle adjacent to a given triangle takes $O(N)$ triangle-search operations. With the adjacency information provided by the TDCEL data structure, the same operation only takes $O(1)$ triangle-search steps.

The surface of the brain tissue and its solid texture were generated by the following procedure from 3D medical data. First, the MRI data were cropped slice by slice, by means of the CliPSS medical-image editor [1], to leave only the region of interest, the area containing the brain tissue. The stack of cropped images was used to generate both a polygonal surface model and a solid texture.

The polygons (triangles) of the brain surface were generated as an isovalue contour surface. Prior to the polygon generation, we used Gaussian low-pass filtering and down-sampling to reduce the amount of 3D MRI image data from roughly $128 \times 128 \times 128$ voxels (for the cropped image) to $32 \times 32 \times 32$. These down-sampled 3D image data were used to generate the surface model of the brain. As an example of the polygonal complexity of the brain data, the model of the brain in Figure 5 contained 7200 triangles, while the scene of the figure in its entirety contained 12000 triangles.

To generate a solid texture, we down-sampled the cropped gray-scale image to $128 \times 128 \times 64$ voxels and added color resembling that of brain tissue. A transformation matrix that related the solid texture to the polygonal brain-surface model above was computed to match the surface and the texture. The amount of data for the solid texture was limited to the number above because of the limitations of the texture memory of the RealityEngine 2 hardware. Note, however, that the reduction in the number of voxels from the cropped MRI image of the brain to the solid texture is only by a factor of 2.

Grabbing

Grabbing refers to "attaching" an object to the 6-DOF tracker held by the user's hand and moving it. It is one of the most popular operation modes in VEs, probably because it resembles the real-world way of manipulating objects. Some customers even demand an accompanying model of a hand that moves in the virtual world, for greater realism (see Figure 5). The grabbing function implemented in the MTVE is simple: It attaches the object that grabs to the object that is grabbed by a rigid-body transformation that is the spatial relation of the two objects when the grab occurs.

As mentioned before, in MTVE, the world is a tree, with the nodes being objects. The hierarchy is a modeling-transformation hierarchy as well. By using this world representation, we can implement grabbing by manipulation of the tree (see **Figure 11**). When an object B (e.g., a hand) grabs an object D (e.g., the skull), D must move with B, and the relation of the two objects, T_{BD}, must remain the same as it was at the moment the grabbing took place. The transformation T_{BD} can be computed by combining the transform T_{WD} from the object D to the world W (the root of the tree) with the *inverse* of the transformation T_{WB} from the object B to the world W [51].

More realistic grabbing may require collision detection of every part of a hand (or even two hands) with the object(s) being grabbed, and detection of gestures of the hand. It could even require computations involving the force of grip and the coefficients of friction. We have not implemented such sophisticated grabbing.

Flying

The flying operation lets the user (with the camera) fly around in the virtual world in the direction indicated by the stylus. This ability for the user to move distances that are large compared to the range of the tracking device is necessary, since the range of the tracking device is often quite limited. For example, for the Polhemus FasTrak, the tracking information is accurate only within a sphere of a few meters' diameter. With such a limited range, the user cannot even walk around the virtual operating table.

This operation translates a node in the object hierarchy tree as if the node and everything below it were on a "magic carpet." In order for the user to fly, the user's eyes and hand must be placed under a node that acts as a magic carpet. (Otherwise, for example, the hand might be left behind as the eyes flew ahead.) The flying operation moves the node attached to the 6-DOF tracker in the direction indicated by the tracker by repeatedly applying a translation vector at each simulation update.

Cutting and digging

Actual surgery to remove a brain tumor involves a wide variety of operations on many types of tissue, each tissue with very complex properties. A realistic simulation of the surgical procedure would require simulation of elastic and nonelastic deformations, tearing, and other behavior at a real-time update rate. We concluded that truly realistic simulation of surgical operations in real time is not possible at this time. We opted to implement a pair of very simple virtual operations, *cut* and *excavate* (*dig*) to simulate operations on the skin, the skull, and the brain tissue.

The virtual cutting operation opens a hole in a surface of the polygonal models of either the skin or the skull. We experimented with two methods of opening holes: One modifies the opacity of the (2D) texture mapped on the surface, and the other modifies the geometry and topology of the polygons that make up the surface. The latter proved better in terms of both simulation performance and visual quality.

As described previously in the typical simulation scenario, the shape and location of the opening are specified by the stylus and a button. The stylus changes its appearance into a virtual scalpel for the skin and a virtual drill for the skull. The same algorithm is used to cut the openings for both the skin and the skull, given the vertices that define the circumference of the holes.

The following explains the steps to create an opening on the surface model of the skull (see **Figure 12**):

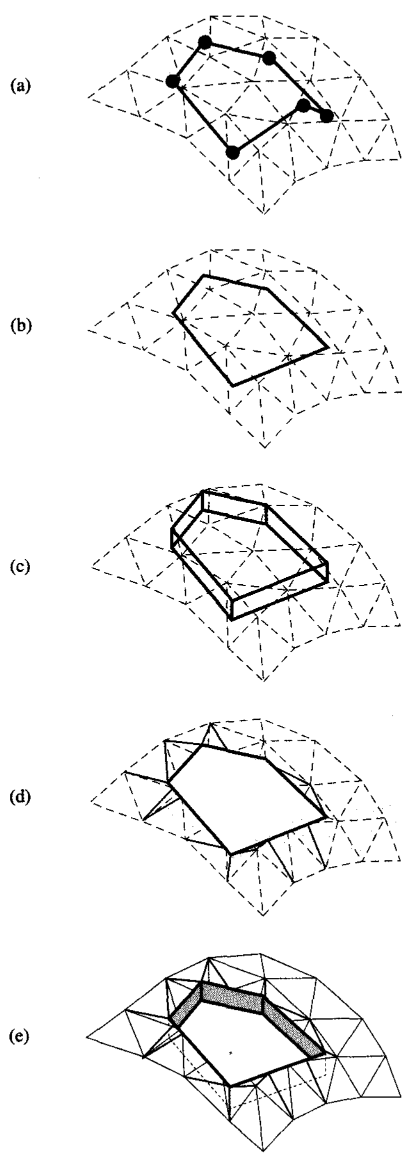

Figure 12

A polygonal opening on the skull is defined by the virtual drill (a stylus with a 6-DOF tracker) by its vertices: (a) holes are drilled at vertices; (b) the opening is made convex; (c) intersections are made; (d) the triangles are re-tesselated; (e) skirts are added.

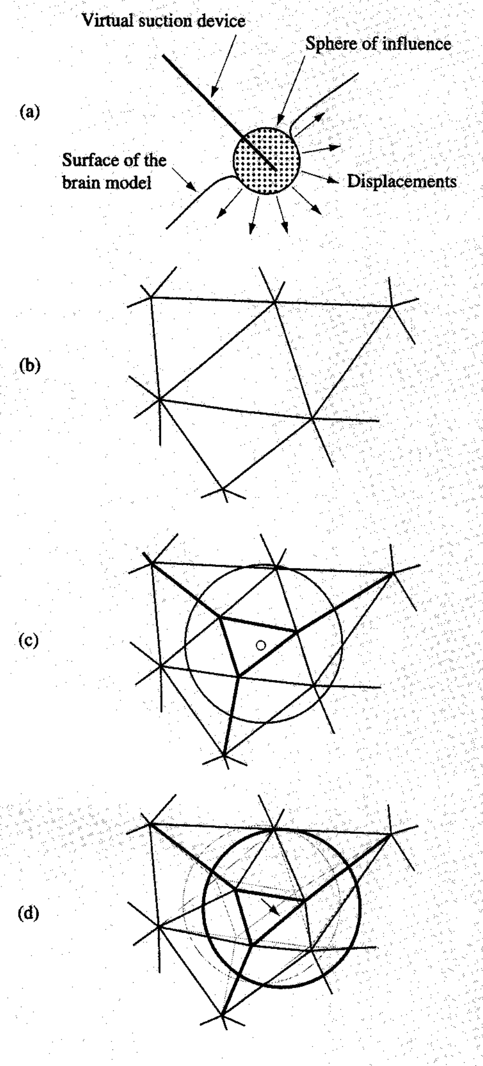

Figure 13

Excavation of the brain tissue by a suction device is simulated by displacing the polygons in the proximity of the tip of the device: (a) sphere of influence; (b) triangles of the hole created by the excavation; (c) adaptive subdivision of triangles to create a smooth pit (edges added are indicated by thicker lines); (d) displacement of triangles in the direction of the suction device.

a. With the virtual drill, mark the vertices of the intended polygonal opening. In order to define the vertices, a line segment in the tip of the drill bit is intersected with the skull surface model.
b. Compute a convex hull from the vertices defined in the previous step, so that the opening is a convex polygon. The convex hull is computed in 2D by projecting the vertices on a plane that is perpendicular to the average of all the vertex normals of the opening.
c. Erect a rectangle on each edge of the convex hull on the plane, and compute the intersections of the plane and the triangles of the skull surface.
d. Re-tessellate the triangles of the surface at the intersections determined in the previous step, while introducing new vertices and edges when necessary. [In Figure 12(d), added edges are shown by solid lines, while original edges are shown by dotted lines.]
e. Add a "skirt" that consists of rectangles to the edge of the opening, so that the perimeter of the opening appears to have a thickness.

In the steps above, traversal of triangles is accelerated with the help of the adjacency information provided by the TDCEL data structure associated with the triangles.

While the example above illustrates the case of creating an opening in the skull, the steps required to cut an opening in the model of the skin are similar. Instead of four to six vertices for an opening in the skull defined by the drill, an opening in the model of the skin is defined by a much larger number of vertices that are captured continuously while the blade of the scalpel is dragged over the model surface.

Virtual digging simulates the removal of tissue from the brain by a suction device. As long as the virtual suction device is touching the brain model, a set of the polygons on the surface of the brain model is displaced repeatedly at each simulation update (**Figure 13**). The displacement occurs as if the surface is pushed in by a sphere. The direction in which the surface is displaced is defined by the orientation of the virtual suction device.

As the triangles are displaced along the surface of the sphere about the tip of the device, the triangles are subdivided adaptively so that the shape of the pit is smooth. The decision on whether or not to subdivide a triangle is based on the ratio of the size of the triangle to the radius of the sphere about the tip of the suction device. The adjacency information provided by the TDCEL data structure is used to quickly find the triangles touched by the sphere of influence that must be subdivided and displaced.

Recall that the brain surface is displayed with a solid texture generated from the MRI image data that were used to produce the brain surface. As the brain tissue is excavated by the virtual suction device, the interior structure of the brain, such as the gray matter and white matter layers, becomes visible. This effect adds to the reality of the visualization.

- *SE subsystem results*

Figures 4 through 7 are snapshots of images from the neurosurgery simulation system. Figure 5 shows the brain surface visualized with the solid texture generated from the MRI data. The menu is used to change the opacity of the objects; in the picture, the skin is almost transparent, while the skull is less transparent. To plan for the surgery, the user would make the brain translucent in order to locate the tumor. The hand may grab the head to rotate and move it, so that its position is most advantageous for the planned surgery. Figure 7 shows a close-up of the head after the skin and the skull have been cut away and a part of the brain tissue has been removed by the suction device to expose the tumor.

The brain model, including the skin, the skull, and the brain surface, consists of approximately 7.2K polygons and a solid texture of size 128 × 128 × 64. The total complexity of the scene in Figures 4 to 7 was 12.1K triangles. The simulator achieved a frame rate of more than ten frames per second for a monocular view on the Onyx/RealityEngine 2 with a single hardware graphics pipeline.

This performance figure may seem less than what might be expected from the specifications of the RealityEngine 2. While the software has much room for performance improvements, there are several performance-limiting factors that are difficult to avoid. For example, performance figures are often quoted assuming the use of the "triangle-strip" primitive, which could significantly reduce the number of vertices that must be transformed, thereby increasing the number of triangles processed per second. However, our system must use individual triangles for the brain and other objects, so that per-triangle collision detection can be performed. Neither does "viewport culling," another optimization that is often useful in other applications, work well in our system, since most of the triangles are visible most of the time.

• *SE subsystem summary*

Our proof-of-concept immersive-VE neurosurgery simulator can visualize the brain anatomy of actual patients from data captured by 3D-medical-imaging equipment. The system then lets users operate on the visualized anatomy by using simplified virtual surgical procedures.

Key features we strove to realize in the system were 1) high-quality visualization of the brain anatomy; 2) simulation of neurosurgery operations; and 3) interactive performance for the visualization and simulation, in an immersive virtual environment. These key features were realized by combining a polygonal surface model with solid texture to depict details for the visualization, and by adding topology to the model by means of the TDCEL data structure for the simulation of surgical operations. We achieved an interactive simulation and visualization update rate (i.e., more than ten frames per second) on a four-processor, single-pipe SGI Onyx/RE2 workstation.

The surgical simulator was written using the MTVE environment, a VE-specific toolkit we developed. The MTVE has functions necessary for building typical virtual-environment application programs. In addition, the MTVE has functions specialized for surgical simulation applications, such as cutting and tissue removal. To define virtual worlds, the MTVE uses VEF files, which define geometries, colors, textures, behaviors, and hierarchies of the objects in the intended virtual world. The neurosurgery simulator and the cancer-information-theater applications were created simply by writing two different VEFs. The MTVE toolkit with the VEF facility proved effective for VE application development. Modifications, such as changing the menu structure, repositioning objects, and changing the object hierarchy, can be done quickly by editing the VEF.

• *Discussion and future work*

The neurosurgery simulator system showed promise for surgical simulation in an immersive virtual environment. However, application of the neurosurgery simulator system to actual planning, optimization, and rehearsal of neurosurgeries on a daily basis would require further improvements.

Many issues that must be solved are common to other immersive VE and AR systems. For example, the system lag is too long, the rendering and simulation update rates are too low, and the HMD is too cumbersome to wear, has a low display resolution, and has a low dynamic range. In addition to these issues in common with other immersive VE and AR systems, there are issues specific to the neurosurgery simulator, which are discussed in the following paragraphs.

The quality of simulation would increase significantly if major blood vessels were added to the model of the brain. Blood vessels are important, since the avoidance of blood vessels is often critical to the welfare of the patient. Acquisition and modeling of the blood vessels from medical images taken of the patients (e.g., by magnetic-resonance angiography) will be the key to solving this problem.

While the visualization method we have used for the brain by using solid texture was quite effective, it was not sufficient. One of the major deficiencies is the lack of proper 3D cues, which is apparent when a pit is created by excavating the brain tissue (Figure 7). While the pit does appear depressed when motion parallax and/or binocular disparity are employed, the 3D cues disappear when the viewpoint is static and the viewing is monocular. Monocular depth cues, such as shadowing and specular highlighting, would be difficult to realize on the current generation of graphics hardware with the scene complexity and rendering update rate requirements of our system. We hope that specular highlighting and shadowing will become standard without significant performance penalty in future generations of graphics engines.

Interaction techniques for surgical simulators in immersive virtual environments must be explored further, especially by adding tactile and kinesthetic feedback. Without such feedback, cutting with the use of the virtual scalpel can be very difficult and imprecise. Tactile and kinesthetic feedback, as well as better visual feedback, requires more detailed modeling and simulation of the physical properties of the skin, brain, and other objects. Such simulation may involve, for example, finite-element-method simulations running on a parallel supercomputer. The alternative to the 3D-menu-based interface, which is often unnatural and awkward, must also be explored.

Another path of exploration is the adoption of an AR interface to guide surgeries, for example by augmenting views of the patient with rendered images of tumors and other anatomical features as well as a planned path to access the tumor. As mentioned before, Kikinis and others [32, 35] have attempted this approach, although they did not display images on the basis of tracked head locations and orientations at interactive speed. Acquisition of shapes and location of real-world objects, e.g., the head, and their registration with virtual objects, e.g., an MRI image of the head, especially in real time, is one of the challenging issues that must be studied [50, 52].

- *Lessons learned*

We have learned a few nontechnical lessons through the development of the neurosurgery simulator system:

- *Selecting the right goal:* Knowing the difficulty of implementing a realistic craniotomy simulator, and given the required development time (ten months from the concept, eight months after the arrival of the hardware), we suggested to the clients (surgeons) the definition of different short-term goals. For example, application of a VE or AR to stereotaxic surgeries could yield a clinically useful system faster. The doctor argued for the challenging craniotomy simulation in a VE. We needed to greatly simplify the simulation so that it could be accommodated to the hardware and software available.

  We failed in defining the right goal. We should have tried harder to educate and persuade our client, so that we could work together for a more realistic and meaningful goal.
- *Selecting the right tool:* When faced with a one-time development project, the developer is often asked the following questions: 1) Is there an off-the-shelf product that can be used, and, if not, 2) is it worthwhile to develop a more-or-less general toolkit? For various reasons, we decided not to use a commercial toolkit designed specifically for the development of virtual-environment applications (for example, [40, 41]). The major reason was the lack of the functionalities necessary for our purpose in such packages, e.g., cutting and deforming objects to simulate surgical operations. Furthermore, they lacked the easy extendibility to add functions of our own. We started off prototyping and testing alternative methods of realizing functions necessary for visualizing 3D medical images and simulating surgical operations. We later started building our own toolkit for the development of VE applications, which incorporated the components developed in the prototypes.

  We made the right decision in building the tool. However, were we to develop a similar surgical simulator today, we would first look harder for commercially available tools.
- *Early involvement with the customers:* Because of the nature of the project, a system integration project, the development process was carried out in a different manner than typical collaborative research (assuming, of course, that "typical collaborative research" exists). We had to follow specifications that were drafted by persons expert on SE, and we did not have the final say regarding many of the issues, including the goal of the VE system development. Consequently, the project included various idiosyncrasies. For example, the hardware system was configured suboptimally, since the components were selected before the manner in which they would be used was known.

  Early communication with clients would have reduced or eliminated this kind of problem; however, such early communication is often forbidden in an open bidding process.

4. Summary

This paper has discussed a system integration project for the National Cancer Center of Japan, with emphasis on a subset, the medical-image system, which includes five subsystems. The paper presented detailed descriptions of the image-database subsystem and the synthetic-environment subsystem.

The image-database subsystem is among the largest and most integrated medical-image database systems yet developed, both in terms of the number of imaging modalities integrated and the daily data volume. The image database handles images not only from computer-friendly sources such as X-ray CT and MRI, but also from X-ray radiography, photomicroscopy, endoscopy, and ultrasound echography equipment. The average data volume from these modalities exceeds 4 GB per day. The database was designed with ease of use in mind so that doctors, nurses, and researchers can use it without much training.

To support a number of modalities, the system handles three types of image data; computer-readable image files, NTSC video signals, and film. We have developed a common image-file format for the database, which serves as a common ground for many formats used by a score of imaging devices and off-the-shelf image-visualization programs. To keep images on-line for a period long enough for clinical purposes, the database has a storage hierarchy that uses 100 GB of hard disks and 900 GB of magneto-optical disks. To facilitate quick transfer of image files between database-access clients and the server, a fast network that uses FDDI was constructed. The image database is accessed from a database front-end, the Clinical View Station (CVS), which runs on workstations located in offices, laboratories, and operating rooms. The CVS provides an integrated, easy-to-use interface for inquiring, retrieving, viewing, and editing images and their attributes.

The synthetic-environment subsystem was intended to explore the applicability of synthetic-environment technology in medicine. We have developed two proof-of-concept applications, the neurosurgery-simulation system and the cancer-information theater. The neurosurgery-simulation system was intended to facilitate planning, rehearsing, and learning about craniotomies for the removal of brain tumors. In an immersive virtual environment, users can visualize 3D medical images acquired from patients by means of medical 3D imaging modalities. The users can then simulate surgical operations such as cutting, drilling, and removal of tissue, albeit in a crude manner. The cancer-information theater is a virtual-environment multimedia anatomy viewer with the emphasis on cancer, intended to facilitate the process of informed consent.

Acknowledgments
This project has been conducted under a contract with the National Cancer Center of Japan. The success of such a large project cannot be achieved without the help of many. We thank Fumihiko Wakao, M.D., NCC Central Hospital, and Ryuzo Sekiguchi, M.D., NCC Hospital East, for their valuable comments and discussions on the image-database system. Credit is also due to Hiroshi Oyama, M.D., NCC Central Hospital, for his valuable suggestions and discussions on the synthetic-environment systems. The authors also thank Mikihiko Ioka, who designed and developed a large part of the image database, and Susumu Suda, for his contributions to the synthetic-environment systems. Finally, we wish to acknowledge many others who worked on the project.

References and notes

1. R. Yoshida, T. Miyazawa, T. Otsuki, and A. Doi, "Clinical Planning Support System—CliPSS," *IEEE Computer Graph. & Appl.* **13,** No. 6, 76–84 (1993).
2. A. Watt and M. Watt, *Advanced Animation and Rendering Techniques*, ACM Press, New York, 1992.
3. *Scientific Visualization—Advances and Challenges*, L. Rosenblum, R. A. Earnshaw, J. Encarnacao, H. Hagen, A. Kaufman, S. Klimenko, G. Nielson, F. Post, and D. Thalman, Eds., Academic Press Ltd., London, 1994.
4. P. M. Keller and M. M. Keller, *Visual Cues—Practical Data Visualization*, IEEE Computer Society Press, Los Alamitos, CA, 1993.
5. ACR–NEMA, "Digital Imaging and Communications in Medicine," NEMA standards publication, Washington, DC, 1994; available from *ftp://ftp.xray.hmc.psu.edu/dicom_docs/dicom_3.0.*
6. O. Ratib, Y. Ligier, and J. R. Scherrer, "PAPYRUS 3.0: DICOM Compatible File Format," *Med. Informat.* **19,** No. 2, 171–178 (1994).
7. *Oracle 7 Server Concepts Manual*, Order No. PN.6693-70.1292, 1993; Oracle Corporation, 500 Oracle Parkway, Redwood Shores, CA 94065 *(http://www.oracle.com).*
8. K. Akeley, "RealityEngine Graphics," *Proceedings of SIGGRAPH '93*, 1993, pp. 109–116.
9. E. Kitagawa, T. Yasuda, S. Yokoi, and J. Toriwaki, "An Interactive Voxel Data Manipulation System for Surgical Simulation," *Proceedings of the 3rd IEEE International Workshop on Robot and Human Communication '94*, Nagoya, Japan, 1994, pp. 204–209.
10. *Virtual Reality—Scientific and Technological Challenges*, N. I. Durlack and A. S. Mavor, Eds., National Research Council, National Academy Press, Washington, DC, 1994.
11. M. Bajura, H. Fuchs, and R. Ohbuchi, "Merging Virtual Objects with the Real World: Seeing Ultrasound Imagery Within the Patient," *ACM Computer Graph. (Proceedings of SIGGRAPH '92)* **26,** No. 2, 203–210 (1992).
12. R. M. Taylor, W. Robinett, V. L. Chi, F. P. Brooks, W. V. Wright, R. S. Williams, and E. J. Snyder, "The Nanomanipulator: A Virtual-Reality Interface for a Scanning Tunneling Microscope," *ACM Computer Graph. (Proceedings of SIGGRAPH '93)* **27,** 128–134 (1993).
13. S. Feiner, B. McIntyre, and D. Seligmann, "Knowledge-Based Augmented Reality," *Commun. ACM* **36,** No. 7, 52–62 (1993).
14. D. Sims, "New Realities in Aircraft Design and Manufacture," *IEEE Computer Graph. & Appl. (CG&A)* **14,** No. 2, 91 (1994).
15. I. E. Sutherland, "The Ultimate Display," *Proceedings of IFIP Congress 1965,* Vol. 2, 1994, pp. 506–508.
16. M. Deering, "High Resolution Virtual Reality," *ACM Computer Graph. (Proceedings of SIGGRAPH '92)* **26,** No. 2, 195–201 (1992).
17. K. W. Arthur, K. S. Booth, and C. Ware, "Evaluating 3D Task Performance for Fish Tank Virtual Reality," *ACM Trans. Info. Syst.* **11,** No. 3, 239–265 (1993).

18. R. S. Kalawsky, *The Science of Virtual Reality and Virtual Environments,* Addison-Wesley Publishing Co., Reading, MA, 1993.
19. G. Burdia and P. Coiffet, *Virtual Reality Technology*, John Wiley & Sons, Inc., New York, 1994.
20. (a) R. H. Taylor, J. Funda, L. Joskowicz, A. D. Kalvin, S. H. Gomory, A. P. Gueziec, and L. M. G. Brown, *IBM J. Res. Develop.* **40,** 163–183 (1996, this issue); (b) R. H. Taylor, B. D. Mittelstadt, H. A. Paul, W. Hanson, P. Kazanzides, J. F. Zuhars, B. Williamson, B. L. Musits, E. Glassman, and W. L. Bargar, "An Image-Directed Robotic System for Precise Orthopaedic Surgery," *IEEE Trans. Robotics & Automation* **10,** No. 3 (1994).
21. R. Ohbuchi, A. State, D. Chen, A. Brandt, C. Tector, M. Bajura, and H. Fuchs, "Seeing the Baby in the Womb—Real-Time Real-Space Visualization of Medical Images Using Virtual Environment Technology," *IPSJ Graphics & CAD SIG Notes* **94,** No. 59, 15–22 (1994) (in Japanese).
22. D. Hon, "Ixion's Laparoscopic Surgical Skills Simulator," *Proceedings of Medicine Meets Virtual Reality II,* University of California at San Diego School of Medicine, 1994; *(http://www.oip.gatech.edu/berc/bercprojects/eye_surg_sim.html).*
23. K. T. McGovern and L. T. McGovern, "The Virtual Clinic, a Virtual Reality Surgical Simulator," *Proceedings of Medicine Meets Virtual Reality II,* University of California at San Diego School of Medicine, 1994; *(http://www.oip.gatech.edu/berc/bercprojects/eye_surg_sim.html).*
24. J. Peifer, M. Sinclair, R. Haleblian, M. Luxenberg, K. Green, and D. Hull, "Virtual Environment for Eye Surgery Simulation," *Proceedings of Medicine Meets Virtual Reality II,* University of California at San Diego School of Medicine, 1994; *(http://www.oip.gatech.edu/berc/bercprojects/eye_surg_sim.html).*
25. M. A. Sagar, D. Bullivant, G. D. Mallinson, P. J. Hunter, and I. W. Hunter, "A Virtual Environment and Model of the Eye for Surgical Simulation," *ACM Computer Graph. (Proceedings of SIGGRAPH '94)* **28,** 205–212 (1994).
26. N. Suzuki and A. Takatsu, "Development of a Computer-Assisted Support System in the Field of Surgical Operation with 3-Dimensional Imaging Technique," *Simulation and Computer-Aided Surgery,* John Wiley & Sons, Inc., New York, 1994, pp. 183–189.
27. L. F. Hodges, R. Kooper, T. C. Meyer, B. O. Rothbaum, D. Opdyke, J. J. de Graaff, J. S. Williford, and M. M. North, "Virtual Environments for Treating the Fear of Heights," *IEEE Computer* **28,** No. 7, 27–34 (1995).
28. R. Kijima, K. Shirakawa, M. Hirose, and K. Nihei, "Virtual Sand Box: Development of an Application of Virtual Environment for Clinical Medicine," *Presence* **3,** No. 1, 45–59 (1994).
29. M. Hirose, M. Taniguchi, Y. Nakagaki, and K. Nihei, "Virtual Playground and Communication Environments for Children," *IEICE (Institute of Electronics, Information and Communication Engineers) Trans. Info. & Syst.* **E77-D,** No. 12 (December 1994).
30. E. Watanabe, T. Watanabe, S. Manaka, Y. Mayanagi, and K. Takakura, "Three-Dimensional Digitizer (Neuro-Navigator): New Equipment of CT-Guided Stereotaxic Surgery," *Surg. Neurol.* **27,** 543–547 (1987).
31. G. H. Barnett, D. W. Kormos, C. P. Steiner, and J. Weisenberger, "Use of a Frameless, Armless Stereotactic Wand for Brain Tumor Localization with 2-D and 3-D Neuroimaging," *Neurosurgery* **33,** 674–678 (1993).
32. W. Lorensen, H. Cline, R. Kikinis, D. Altobelli, and L. Gleason, "Enhancing Reality in the Operating Room," *Proceedings of IEEE Visualization '93*, San Jose, CA, October 1993, pp. 410–415.
33. B. L. K. Davey, R. M. Comeau, P. Munger, L. J. Pisani, D. Lacerte, A. Olivier, and T. M. Peters, "Multimodality Interactive Stereoscopic Image-Guided Neurosurgery," *Proceedings of Visualization in Biomedical Computing '94, SPIE Proc.* **2359,** 526–536 (1994).
34. A. C. F. Colchester, J. Zhao, R. L. Evans, P. Roberts, N. Maitland, D. J. Hawkes, D. L. G. Hill, A. J. Strong, D. G. Thomas, M. J. Gleeson, and T. C. S. Cox, "Craniotomy Simulation and Guidance Using a Stereo Video Based Tracking System (VISLAN)," *Proceedings of Visualization in Biomedical Computing '94, SPIE Proc.* **2359,** 541–551 (1994).
35. R. Kikinis, P. L. Gleason, W. Lorensen, W. Wells, W. E. L. Grimson, T. Lozano-Perez, G. Ettinger, S. White, and F. Jolesz, "Image Guidance Techniques for Neurosurgery," *Proceedings of Visualization in Biomedical Computing '94*, Rochester, MN, 1994, pp. 537–540.
36. R. A. Robb, "Virtual Reality Assisted Surgery Program," *Proceedings of Medicine Meets Virtual Reality III,* University of California at San Diego School of Medicine, 1995; *(http://www.oip.gatech.edu/berc/bercprojects/eye_surg_sim.html).*
37. D. R. Peachey, "Solid Texturing of Complex Surfaces," *ACM Computer Graph. (Proceedings of SIGGRAPH '85)* **19,** 279–286 (July 1985).
38. K. Perlin, "An Image Synthesizer," *ACM Computer Graph. (Proceedings of SIGGRAPH '85)* **19,** 287–296 (July 1985).
39. C. Codella, R. Jalili, L. Koved, and J. Lewis, "A Toolkit for Developing Multi-User, Distributed Virtual Environments," *Proceedings of the Virtual Reality Annual International Symposium (VRAIS) '93*, Seattle, WA, 1993.
40. C. Shaw, M. Green, J. Liang, and Y. Sun, "Decoupled Simulation in Virtual Reality with the MR Toolkit," *ACM Trans. Info. Syst.* **11,** No. 3, 287–317 (1993).
41. S. Ghee, "Programming Virtual World," *ACM SIGGRAPH '94 Course Node No. 17*, Orlando, FL, 1994, pp. 6-1–6-58.
42. SENSE8 "WorldToolKit TECHNICAL OVERVIEW," 1994 sales literature, SENSE8 Corporation, 6000 Bridgeway, Suite 104, Sausalito, CA 94965 *(http://www.sense8.com).*
43. J. Rohlf and J. Helman, "IRIS Performer: A High Performance Multiprocessing Toolkit for Real-Time 3D Graphics," *ACM Computer Graph. (Proceedings of SIGGRAPH '94)* **28,** 381–394 (1994).
44. J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes, *Computer Graphics: Principle and Practice*, Addison-Wesley Publishing Co., Reading, MA, 1990.
45. H. Samet, *The Design and Analysis of Spatial Data Structures*, Addison-Wesley Publishing Co., Reading, MA, 1989.
46. A. State, J. McAllister, U. Neumann, H. Chen, T. Cullip, D. Chen, and H. Fuchs, "Interactive Volume Visualization on a Heterogenous Message-Passing Multicomputer," *Proceedings of the 1995 Symposium on Interactive Computer Graphics*, ACM, Monterey, CA, 1995, pp. 69–74.
47. T. J. Cullip and U. Neumann, "Accelerating Volume Reconstruction with 3D Texture Hardware," *Technical Report TR93-027*, Department of Computer Science, UNC-Chapel Hill, NC, 1993.
48. T. Yoo and D. Chen, "Interactive 3D Medical Visualization: A Parallel Approach to Surface Rendering 3D Medical Data," *Computer Applications to Assist Radiology (Proceedings of the Symposium for Computer Assisted Radiology 94)*, J. Boehme, A. Rowberg, and N. Wolfman, Eds., Winston-Salem, NC, 1994, pp. 100–105.
49. W. E. Lorensen and H. E. Cline, "Marching Cubes: A High Resolution 3D Surface Construction Algorithm," *ACM Computer Graph. (Proceedings of SIGGRAPH '87)* **21,** No. 4, 163–169 (1987).
50. F. P. Preparata and M. I. Shamos, *Computational Geometry—An Introduction*, Springer-Verlag, New York, 1985.

51. W. Robinett and R. Holloway, "Implementation of Flying, Scaling, and Grabbing in Virtual Worlds," *Proceedings of the 1992 Symposium on Interactive 3D Graphics*, ACM, Cambridge, MA, 1992, pp. 189–192.
52. D. A. Simon, M. Hebert, and T. Kanade, "Techniques for Fast and Accurate Intrasurgical Registration," *J. Image Guided Surgery* **1,** 17–29 (1995).

Received April 10, 1995; accepted for publication September 22, 1995

Ryutarou Ohbuchi *Mobile and Graphics Technology, Tokyo Research Laboratory, IBM Japan Ltd., 1623-14 Shimotsuruma, Yamato-shi, Kanagawa-ken 242, Japan (ohbuchi@trl.ibm.co.jp).* Dr. Ohbuchi received the B.S. degree in electrical and electronic engineering from the Sophia University in Tokyo in 1981, the M.S. degree in computer science from the University of Electrocommunications in Tokyo in 1983, and the Ph.D. degree in computer science from the University of North Carolina at Chapel Hill in 1994. His research interests include augmented reality systems, scientific visualization, realistic image synthesis, medical imaging and graphics, human–computer interaction, and parallel computer architecture. Dr. Ohbuchi is a member of the IEEE Computer Society, the ACM, and the Information Processing Society of Japan (IPSJ).

Tatsuo Miyazawa *Network and Solution Technology, Tokyo Research Laboratory, IBM Japan Ltd., 1623-14 Shimotsuruma, Yamato-shi, Kanagawa-ken 242, Japan (MIYAZAWT at TRLVM; miyazawt@trl.ibm.com).* Mr. Miyazawa is a researcher at the Tokyo Research Laboratory, IBM Japan. He received B.E. and M.E. degrees in aeronautical engineering from the University of Tokyo in 1982 and 1984, respectively. He joined IBM Japan in 1984, transferring to the Tokyo Research Laboratory in 1988. Mr. Miyazawa's research interests include volume visualization, medical imaging, and virtual reality. He is a member of the IEEE, the IEEE Computer Society, ACM SIGGRAPH, and IPSJ.

Masaki Aono *Mobile and Graphics Technology, Tokyo Research Laboratory, IBM Japan Ltd., 1623-14 Shimotsuruma, Yamato-shi, Kanagawa-ken 242, Japan (aono@trl.ibm.co.jp).* Dr. Aono received B.S. and M.S. degrees in information science from the University of Tokyo in 1981 and 1984, and the Ph.D. degree in computer science from Rensselaer Polytechnic Institute in 1994. His research interests include computer-aided geometric design, physically and biologically based modeling, photo-realistic rendering, differential and computational geometry, computer animation, and virtual environment for medical applications. Dr. Aono is a member of the IEEE Computer Society, the ACM, TUG (Tex Users' Group), IPSJ, and the ISO/IEC JTC1/SC24 standardization technical committee on CGM (Computer Graphics Metafile).

Akio Koide *Network and Solution Technology, Tokyo Research Laboratory, IBM Japan Ltd., 1623-14 Shimotsuruma, Yamato-shi, Kanagawa-ken 242, Japan (koide@trl.ibm.com).* Dr. Koide received his Doctor of Philosophy degree in physics from the University of Tokyo in 1975; he worked as a research associate at the University of Western Ontario, Canada, before joining IBM in 1981. Dr. Koide currently manages the Exploratory Network group in the laboratory and works in the area of network security, copyright protection, and real-time 3D user interfaces.

Masahiko Kimura *System Integration No. 4, Health System Development Office, Solution Operations, IBM Japan Ltd., 19-21, Nihonbashi Hakozaki-cho, Chuo-ku, Tokyo 103, Japan (e23527@tokvmic2.vnet.ibm.com).* Mr. Kimura received his M.Eng. degree in astronautics from the University of Tokyo in 1990. He was a researcher at the IBM Tokyo Research Laboratory during the NCC project. In January 1996, he moved to the Health System Development Office in IBM Japan Ltd. His current research interests include techniques for anatomical modeling and its application to surgical planning.

Ryo Yoshida *Network and Solution Technology, Tokyo Research Laboratory, IBM Japan Ltd., 1623-14 Shimotsuruma, Yamato-shi, Kanagawa-ken 242, Japan (yoshidar@trl.ibm.co.jp).* Mr. Yoshida received his B.Eng. and M.Eng. degrees in mechanical engineering from the Tokyo Institute of Technology, Japan, in 1988 and 1990, respectively. He has since been a researcher in the medical imaging group at the IBM Tokyo Research Laboratory. His research interests include VR applications, 3D imaging, and image processing in medicine. Mr. Yoshida is a member of the IEEE and SPIE.