Interactive Exploration in Virtual Environments

Belleman, R.G.

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
Chapter 2

Design consideratons for interactive exploration environments*  

"Good design comes from experience, experience comes from bad design."

Frederick P. Brooks, Jr.

2.1 Introduction

In this chapter we investigate the different issues that are involved in the development of an exploration environment that allows interactive exploration of large data and parameter spaces. We distinguish three different aspects in such a system: the available technology, the applications and the scientific issues.

2.1.1 Technological issues

The technological aspects are mainly concerned with the computational hardware that is used for the exploration system. The increase in capabilities of modern computer systems has been impressive, in some cases even allowing both the computation and presentation tasks to be executed on the same machine. However, a performance increase may be attained by running these tasks on dedicated machinery. For example, many simulation applications perform better on dedicated hardware such as vector processors, massively parallel platforms or other high performance computing systems. Also, state-of-the-art graphical systems are now available that are well suited for the presentation tasks. When the simulation and presentation tasks are

distributed over different systems, some means of communication is required between the two.

An important goal towards a successful exploration environment is to involve the researcher in the presentation as much as possible, thereby increasing the researcher's level of awareness [29]. To achieve this, an exploration system needs the following, often conflicting capabilities:

- **Quality of presentation** – The most common method to provide insight in simulated phenomena is to represent the abstract data as visual geometric constructs that present quantitative and relational aspects to the observer in an intuitive manner. Many scientific visualization techniques are now available that provide means of efficiently achieving this [209, 242]. However, for some applications mere visual exploration will not always be sufficient. These applications can benefit from the integration of additional sensory modalities to increase the user's field of perception. Additional modalities applied in previous research include the use of sound [68, 93, 141], touch to provide “haptic feedback” or “tactile feedback” [28, 33, 36, 65, 66, 91, 109, 153, 224] and even smell [125, 138].

- **Rapid frame rate** – While the capabilities of modern graphical workstations allow the construction of high quality and complex images with relative ease, the level of detail in the presentation should be minimized to avoid information clutter and to achieve high frame rates (the aim should be to employ “minimal means for maximum effect”). For a usable exploration environment the visual frame rate should be at least 10 frames per second. Insufficiently high frame rates can lead to disorientation and even nausea, a phenomenon which is often called “cybersickness” in the context of virtual environments and is akin to motion sickness [131, 156]. Note the difference between “frame rate” and a display's “refresh rate” which is the frequency at which a display device redraws an image on screen. It is generally accepted that a display refresh rate of more than 60 Hz results in a stable, flicker free image. Also note that a frame rate that is higher than a display's refresh rate results in wasted computing power.

- **Intuitive interaction** – Some level of interaction with a presentation is mandatory. Unlike the standard interaction metaphors used in windowing systems on workstations, no standard user interaction metaphors yet exist for virtual environments. Most notably in the “living simulation model”, the increase in functionality expected from an exploration environment demands a well considered user interface.

- **Real-time feedback** – Some delay will always occur between the moment a user interacts with a presentation and the moment that a response is available. This is caused by low tracking rates of input devices, communication delay between the exploration and the simulation system and temporary reduced availability of computational or network resources. To attain accurate control over a running application, the total amount of lag should not exceed a couple of seconds. If
longer, users tend to think that their interaction was not recognized and should be repeated or that the system is defective [171].

2.1.2 Application issues

Each application has different execution characteristics with respect to the update frequency, the point in the application where interaction may occur and at what moment, and the amount of data that changes in between each update and thus needs to be communicated to the presentation system.

The relative importance of each of these capabilities depends primarily on the characteristics of the simulation application and the limits imposed by the available infrastructure. It is our intention to develop a conceptual model with which we can describe the behaviour of an exploration system. To obtain an accurate representation of the behaviour of the system as a whole, this model should include the temporal characteristics of the application, the presentation software, the interconnection between the two, and the hardware each runs on.

2.1.3 Scientific issues

The key issue towards a successful exploration environment is real-time interaction [29]. One way to achieve this is through a trade-off between speed and accuracy of the exploration by allowing "short-cuts" to be made in a guided search by an interactive feedback mechanism between the application and the exploration system. The scientific question we want to address is concerned with the implications of making these "short cuts".

The guided search algorithms that are used to search a problem's parameter space are often based on methods that stem from mathematical, biological or physical models at a macroscopic level. To be able to perform computer simulations, these macroscopic models must first be converted into a computable algorithm. At this microscopic level, it is expected that the computable algorithm provides a sufficiently accurate abstraction from the macroscopic model such that it achieves the desired result. However, the process of making "short cuts" on a microscopic level may have severe implications on the model that is being simulated. Indeed, the resulting simulation may well be totally different from that described at the macroscopic level.

2.1.4 Test cases

As an initial assessment of the available technology and to obtain a better understanding of the application and scientific issues involved in interactive exploration systems, we built a number of static environments for applications from different research areas.
2.2 Car crash simulation playback

The PAM-CRASH package developed by the ESI Group is a numerical simulation package for prototyping and manufacturing processes that take into account the temporal mechanics of car crashes at high accuracy [75]. The ESI Group's product portfolio provides a virtual engineering solution known as the "Virtual Try-Out Space" (VTOS). This solution allows reduced costs and development lead times by progressively eliminating the need for physical prototypes. The PAM-CRASH software is based on a Finite Element simulation model. Depending on the desired accuracy, simulations can take days to complete on even the most powerful computer. In the ESPRIT "CAMAS" project the Section Computational Science of the University of Amsterdam investigated typical data dependencies in order to parallelize the PAM-CRASH code [58]. PAM-CRASH was also used as a test-bed in the Esprit project "Dynamite" where it was investigated how to use computing cycles on idle workstations in an organization through dynamic load balancing [44,243].

When a simulation run is complete, the data is visualized on a desktop computer system for inspection. The ESI Group was interested in knowing whether a VR device like the SARA CAVE would give additional benefits to the visualization of simulation results compared to desktop methods.

2.2.1 Implementation

A number of PAM-CRASH simulation runs were performed by the ESI Group and the resulting data files were gathered for visualization in the SARA CAVE. Each data set consists of multiple files, one for each time step, with the number of time steps varying between tens to close to one hundred. Each file contains a description of the simulated object stored as quadrilateral patches consisting of four vertices and a normal vector for each vertex. We designed and implemented a program that allows the simulation results to be played back in a virtual environment (see also Figure 2.1). The program is implemented in C [126], uses CAVElib [249] for control over the VR equipment (in this case the displays and interaction devices in the SARA CAVE) and OpenGL for graphics rendering [16,189,261]. In this case, OpenGL uses the vertex and normal information to render the object under simulation as a smooth shaded surface. To allow for interactive playback of the simulation results, all time steps are read in at program startup and compiled into OpenGL "display lists" [261]. A display list consists of one or more OpenGL commands that are compiled into a format that can be executed more efficiently by the OpenGL rendering engine. In general, the time required to execute a display list is significantly lower than the sum over the execution time of its constituents. Therefore, the use of display lists increases frame rate which in turn increases responsiveness of the system as a whole. This design choice means that some time is spent between program startup and the moment the user can start using the program but since this time never exceeds approximately one minute, even for the largest data sets, this was considered acceptable.

After initialization, the VE represents the first time step of the simulation. In ad-
2.2 Car crash simulation playback

Figure 2.1: Car crash simulation playback in the CAVE (see also colour reproduction on the back cover).

dition, a virtual pointing stick is drawn from the front of the wand, represented by a simple line, which can be used by the user to indicate points of interest to fellow viewers. This simple mechanism turns out to be an adequate solution to resolve the problem that exists in most stereoscopic projection based VR systems, namely that the perceived location of a virtual object as seen by the tracked viewer differs from that perceived by a guest viewer (as previously described in section 1.5.2 on page 19). Using the three buttons on the CAVE wand, the user plays back the crash simulation much in the same way as the “fast forward” and “rewind” buttons on a video cassette recorder (VCR); while pressed, the right button plays the simulation forward in time, the left button backwards in time. Pressing the middle button replays the simulation from start to finish. The wand’s joystick is used to “navigate” through the VE. Navigation provides users with methods to move beyond the confines of the CAVE’s physical dimensions. Objects beyond the CAVE walls come into reach by moving the CAVE towards them. Note how this concept places the user of the VE in the center of this type of interaction; the user is transported from one place to another while the objects remain where they are. As shown in Figure 2.2, pressing the joystick forward moves the user towards the direction in which the wand is pointing, pulling the joystick backwards moves the user away from the direction in which the wand is pointing. Pressing the joystick sideways rotates the view left or right.

2.2.2 Experiences

Although the information in the data sets only contains a description of the deforming structure over time, the program can easily be extended to visualize any additional
information that may have been calculated during the simulation, such as, for example, stress analysis results.

From a qualitative standpoint, the motion parallax effect provides a very intuitive way to inspect virtual objects. Where previously the user had to use an indirect interaction method (such as a mouse or keyboard) to orient the object on a desktop system, in the VE the user simply moves his viewpoint by walking around the object or moving the head. The navigation methods, controlled by the wand’s joystick, are used to manoeuvre the CAVE through the virtual world to a location of interest. After that, the user simply walks around within the confinements of the CAVE for closer inspection. With some exercise, this interaction method is adequate for moving the viewer to any location of interest in little time.

The interaction requirements for this application were very simple. The biggest shortcoming we found with this application is its lack of expressiveness. For example; with the current implementation the “fast forward/rewind” type of interaction is not a practical interaction method to wind the visualization to one particular time step. The user needs to let go of the button precisely at the desired time step, which is easily missed if the animated playback is too fast. Of course, this could have been solved by having the left and right wand buttons advance only one time step at each button press, but that would then make it difficult to get a clearer idea on the behaviour of the simulation over longer periods of time. Again this problem can be solved, possibly by mapping short button presses to single time step advances and longer button presses to animated time step advances, or perhaps combinations of buttons pressed at the same time, etc., but imagine having to explain all this to the end-user. Moreover, the implementation of these interaction methods very quickly becomes far from trivial.

Clearly the user would benefit from interaction methods that allow more expressiveness than provided by the wand’s buttons and joystick. The most obvious would be to extend the application with something that is similar to the graphical user interface (GUI) interaction methods we know well from desktop systems, i.e. menus, buttons, sliders, etc. Unfortunately, very few GUI toolkits exist for VR applications.
and the ones that do exist are often very limited in their capabilities. Nevertheless, the availability of such a GUI toolkit would be a powerful way to obtain user-friendly and intuitive exploration environments. We will present a solution for this in section 4.3 (page 75), which describes a software architecture that allows existing 2D GUI toolkits to be used in VEs.

2.3 The Virtual Radiology Explorer

The use of VR technology in medical applications has already shown great potential in past studies [166, 197, 273, 274]. The applications developed in these studies have allowed a better understanding of complex anatomical structures. This has helped in areas such as clinical diagnosis, treatment planning, simulation and surgical intervention. However, a larger scale introduction into the medical society has met with resistance for various reasons. In this section we will look into the current methods that are used by radiology departments for the inspection of medical scans for diagnostic purposes and will investigate the use of VR technology in this respect. The Virtual Radiology Explorer (VRE) project\(^1\) was initiated to provide radiologists and physicians with a system that allows them to explore three-dimensional (3D) medical data sets, such as computed tomography (CT) and magnetic resonance imaging (MRI) scans, using VR techniques. One of the aims in this project is to provide intuitive, responsive and suitable interaction techniques through which the end-users will be able to efficiently perform diagnostic tasks.

2.3.1 The radiology department

To create an inventory of current methods that are used to interact with CT and MRI data sets, two radiology departments were involved: one at the University Hospital Utrecht (AZU, headed by prof. dr. P.F.G.M. van Waes), the other at Leiden University Medical Center (LUMC, headed by prof. dr. J.L. Bloem). Over a period of several days, both departments showed the current methods used to inspect CT and MRI scans for clinical diagnostic purposes, their use of 3D imaging techniques, and shared their ideas on future prospects in this regard. The radiology departments at AZU and LUMC use both CT and MRI scanners as their primary diagnostic instruments in cases where a correct diagnosis requires insight in 3D anatomy. The scans produced by most scanners consist of two-dimensional “slices” that each contain $512 \times 512$ gray scale pixels, each 16 bits in resolution. The number of slices made per scan depend on the spacing between slices, the thickness of each slice and the size of the structure of interest. Spatial resolution is measured by the maximum spatial frequency by which two lines close to each other in a line-bar pattern can be distinguished and is expressed as “line pairs per millimeter” ($\text{lp mm}^{-1}$). The spatial resolution for CT

\(^1\)Funded by the High Performance Computing in the Netherlands (HPCN) Platform under ICES-KIS-2.
scanners is 1 lp mm$^{-1}$, for MRI 0.5 lp mm$^{-1}$ (1998). The spatial resolution of film used in X-ray radiographs is 100 lp mm$^{-1}$ [77].

Both departments transfer the scans both to a hardcopy machine, where high-resolution transparency films are printed, and to a cluster of networked workstations (in both departments Philips *EasyVision* systems are used, running on Sun workstations). Along with other methods used for clinical diagnostics, a physician makes a diagnosis and proposes a treatment for a patient based on the images produced from these scans. Hardcopies of the scans on transparent film is the first and foremost method used for inspection. The hardcopy films are inspected using lightboxes in cases where one or two radiologists need to discuss a patient (see Figure 2.3). There is talk in both hospitals to switch over to “filmless” radiology departments, i.e. to a situation where only digital workstations are used for the storage and inspection of medical scans.

Beside the scanned image, the films contain information on the parameter settings of the scanner (including acquisition parameters), patient name, scan date, slice index and annotations that provide a reference for the radiologist on the orientation of the slices. AZU also uses overhead camera systems that enlarge specific regions of the hardcopy onto a television screen so that more people can see and discuss the images at the same time. Results of the inspection are voice recorded on quick-access mini audio tapes that are filed with the hardcopies into patient dossiers. These tapes are then sent to a department were they are converted into written text by typists or, in some cases, by speech recognition systems. For additional annotation, radiologists use red pencils to highlight regions directly on the film. Another method used for annotation is spoken-word-to-tape to refer to specific slices by index numbers and names of anatomical structures contained therein. The lightboxes often contain more than one set of films pertaining a certain patient. The material that is to be discussed is prepared by assistants on separate frames of the lightbox. These are brought into view by the radiologist by keying in a frame number on a keypad.
2.3.2 Desktop visualization

In cases where a side-by-side display of scanned slices is not adequate, Philips' EasyVision workstations provide a so-called “cine-loop” capability which allows radiologists and physicians to “flip” through a scan using a slider, as if it were a deck of cards. The slices can be displayed in one of three orthogonal orientations: sagittal (any vertical plane that divides the body into left and right parts), frontal (any vertical plane that divides the body into anterior [front] and posterior [back] parts) or transversal (any horizontal plane that divides the body into superior [towards the head] and inferior [towards the foot] parts). A user-defined oblique orientation is obtained by drawing lines on the slices to denote the position and orientation of intersection planes. Beside this cine-loop functionality, the scans can be displayed side-by-side as on the hardcopy films. In this case the number of slides in a row is set interactively, allowing the slices to be viewed at different magnification, as well as with user settable contrast and brightness of the displayed images. The EasyVision workstations are not used for annotation or reporting purposes. The cine-loop feature of the EasyVision workstation is the most often used function. Although the software provides 3D rendering (see Figure 2.4), both volume and surface rendering, these capabilities are used sparsely because radiologists find them too complicated to use and, as a consequence, feel it takes too much time to produce useful results. At AZU, 3D rendering is often delegated to the local imaging research group (which has much experience with 3D visualization), while at LUMC EasyVision is used more for this purpose.

The EasyVision system provides several tools to process a scan before it is visualized in 3D. One such tool allows irregularly shaped areas in a stack of slices to be discarded.
from the 3D rendering so that specific anatomical structures can be highlighted. In addition, Philips provides several extensions for the EasyVision system that provide virtual “walkthrough” capabilities (for example, see Figure 2.5).

2.3.3 VRE objectives

Clearly, three-dimensional representations of medical scans are not common practice in present day radiology departments, despite the fact that CT and MRI scans are inherently volumetric. There is a number of reasons for this. First; the technology to reconstruct these data sets to 3D representations in a timely fashion, of sufficient quality that can also be manipulated interactively has just recently become available at an affordable price. Indeed, both departments have desktop workstations that allow them to create 3D representations of anatomical structures. Little use is made of this, however, mostly because the radiologists do not know how to use the workstations and because it takes too much time to obtain acceptable results. However, there is a strong interest to use the 3D information that is available, as (for example) illustrated by liver surgeon Rory McCloy in Nature, March 2002: “I spend my life looking at 60 slices of salami. [...] I’m trying to do a 3D operation with 2D images” [180].

More important, however, is that the two-dimensional diagnostic methods that are currently used work and therefore there is little desire with the radiologists to use new methods that have not yet proven their value. Therefore, if a paradigm shift such as proposed by VRE is to succeed, the transition for its users should be made as easy as possible. Interaction should therefore be a key concept in its design.

2.3.4 Visualization and interaction methods in VRE

The visualization and interaction methods for VRE should make the transition for physicians and radiologists from their conventional diagnostic methods to a virtual environment as comfortable as possible. The interaction mechanisms described here are primarily intended for use in immersive projection based VR systems such as the CAVE and ImmersaDesk. However, most of the methods described here should still be applicable to other types of VR systems. Additional comments on differences and potential problems are explicitly noted where appropriate. To make the transition for users of VRE as simple as possible, different levels of functionality should be offered, ranging from the “conventional” and “well known” methods that are currently in use, to new methods as proposed by the VRE project. As a basic functionality, VRE should therefore offer radiologists functions that mimic the lightbox and cine-loop as are used in daily clinical diagnostic tasks. Beside that, the new functions that VRE can offer are 3D techniques with new capabilities such as stereoscopic rendering, virtual endoscopy, virtual colonoscopy and others.
2.3 The Virtual Radiology Explorer

Volume visualization

Visualization methods for the representation of 3D scalar lattice volumes are usually separated into two groups: (direct) volume rendering and isosurface modeling [69, 72]. Volume rendering is a technique that is based on ray casting [79]. All scalar values in the 3D lattices are traversed and treated as volumetric elements that contribute both colour and opacity to a virtual ray of light that travels through the data set towards the viewer's eyes [146, 240]. The contribution of colour and opacity is defined through a transfer table that maps a scalar value onto a colour and opacity value. Using this table, structures that are not of interest to the user can be made transparent by mapping its associated scalar values to translucent opacity values. Likewise, interesting structures can be emphasized by mapping the associated scalar values to brighter colours. The definition of a transfer table that yields good visual results can be time consuming, or worse; close to impossible. In most cases, the transfer table is used to define a colour and opacity gradient over a range of scalar values. The assumption is that structures of interest consist of scalar elements that have neighbouring values. In the case of medical data sets, however, this assumption depends on the data acquisition method that was used in obtaining the data set. For example; in CT scans the scalar values (or “Hounsfield units” as they are called in the case of CT) are a representation of the attenuation of an X ray beam through the human body [77]. As this value varies with the density of the tissue (the denser the tissue, the higher the value), similar tissue structures will yield similar values. Defining a transfer table that accentuates the different types of tissue is, in this case, relatively trivial. In MRI scans, however, the scalar values represent the radiofrequency (RF) energy emitted by the nuclei of hydrogen atoms (free or attached to other molecules) in a strong magnetic field after excitation by a microwave radio signal [182]. As most parts of the human body consist of fat and varying concentrations of water (and because these molecules contain hydrogen atoms), the scalar values in the data set are mainly a representation of concentrations of hydrogen. Similar concentrations of hydrogen are not always part of the same physiological structure. This makes the definition of a transfer table far more difficult.

In isosurface modeling, an intermediate representation is first computed that consists of geometric primitives; usually triangles [151]. These triangles represent surface patches through lattice elements of the same value, resulting in a constant value contour surface. As with volume rendering, the definition of this constant, so-called “threshold” value relies on the property that scalar elements that are part of the same structure are of approximately the same value. Because isosurfaces only represent structures at the same scalar value, it can in some cases be difficult to relate the isosurfaces to the structure as a whole. To compensate, multiple isosurface models are often computed at different threshold levels that are then rendered together in the same scene, with different colour characteristics to be able to tell them apart and different opacity settings to reveal otherwise occluded structures.
Interactive volume visualization

Additional challenges in successfully applying these visualization methods in an interactive virtual environment are that (1) the time required to compute the visualization is small enough that the user can change parameters of the visualization method and see the results quickly, and (2) the time required to render the resulting visualization is fast enough to allow interactive exploration in a VE. The first challenge is most apparent with isosurface modeling where an intermediate triangle representation needs to be calculated before this representation can be rendered. The second challenge is determined by the performance characteristics of the graphics hardware.

In the case of volume rendering, the 3D lattice structure has to be completely traversed each time the position or orientation of the user changes and/or when the user changes the transfer table. Various implementations of specialized software and hardware have been developed that allow volume rendering to be used in virtual environments at interactive speeds [37, 102, 133, 191]. Most of these methods use 2D or 3D texture mapping and transfer table lookup techniques that are often accelerated in hardware. Using these techniques, the resulting images can be rendered directly based on the current position and orientation of the user and the defined transfer table. A limiting factor on rendering performance is the “fill rate” of the graphics hardware; the rate at which pixels can be drawn into screen memory. Fill rate is usually measured in millions of pixels per second (Mpixels/s) and is directly dependent on the hardware architecture (i.e. the bandwidth of the memory bus and the ability of the graphics hardware to saturate this bandwidth). In particular in the case of volume rendering, fill rate limits the frame rate when the area that is covered by the resulting image on screen increases.

The intermediate triangle representation used in isosurface rendering is independent of the position of the user and therefore only needs to be calculated once when the threshold value has been set or changed. The triangle representation must then be rerendered for the current position and orientation of the user. However, the time required to render a new frame is directly related to the number of triangles in the isosurface. The number of triangles that a hardware graphics interface is capable of rendering in one second is a popular (but inconclusive) measure to characterize its performance. For example; the InfiniteReality2 graphics pipeline used in the SARA CAVE is capable of rendering 11 million triangles per second [217]. Suppose the virtual environment needs to maintain a frame rate of at least 20 stereoscopic frames per second for a particular application. This means there is only \( \frac{1}{40} \)th of a second to render each frame which in turn implies there should be no more than 275,000 triangles in the complete scene. Although this may seem like much, consider that it may often be necessary to visualize multiple isosurfaces, at different threshold levels, so that anatomical structures can be viewed in relation to others. In these cases it may be necessary to reduce the number of triangles in an isosurface but only if the original geometry can be maintained. Decimation is one technique to reduce the number of triangles in an isosurface triangle mesh while preserving the original topology and forming a good approximation to the original geometry [210].
Although decimation helps in obtaining an isosurface contour that can be rendered at interactive speeds, the additional time required to compute the isosurface can be substantial. Considering that the calculation of an intermediate triangle representation for isosurface rendering operates on neighbouring lattice sites, the total execution time can be decreased by decomposing the 3D lattice into subdomains and perform the isosurface extraction in parallel, on multiple processors [202].

### 2.3.5 Implementation

The architecture of the VRE application is shown in Figure 2.6. VRE is implemented in C++ [227] and runs on all Unix operating systems that support CAVELib and OpenGL. CAVELib is used to hide the intrinsic details of the display technology (such as the placement of the projection displays) and interaction devices (wand, joystick and buttons) [249]. OpenGL is used as the graphical rendering library [261]. For volume rendering we use Silicon Graphics' OpenGL Volumizer, a specialized library that supports hardware accelerated volume rendering on Silicon Graphics hardware\(^1\) [191]. Surface modeling is performed by the Visualization Toolkit (Vtk) [106, 209]. The volume and surface representations can be individually clipped to hide parts of

---

\(^1\)This limits the use of volume rendering to Silicon Graphics systems only. The remaining functions can still be used on other systems. "CAVORE", a CAVE volume rendering package developed by Anton Koning (SARA) for the VRE project, can be used on all systems that support 3D texture mapping [133].
the renderings. This allows hybrid representations that consist partly of a volume rendering and for the other part of a surface rendering.

A simple networked database interface has been implemented that emulates a Picture Archiving and Communication System (PACS) as is used in most radiology departments to interface medical scanners to visualization front-ends. The database contains patient data, stored on a remote IBM SP2 system located at SARA, which is accessed by a user interface from within VRE. VRE and the PACS server communicate using PVM [229]. Beside a database function, the PACS component is also equipped with a computing function that is capable of performing parallel isosurface modeling, which will be described below. If the user wishes to visualize an isosurface representation of a specific dataset, the data set identifier and the desired threshold level are sent to the computing engine which then extracts the isosurface contour and sends the results back to the VRE application in the form of a geometrical representation.

**Parallel isosurface modeling**

The parallel isosurface modeling has been implemented using Vtk. The visualization pipeline is shown in Figure 2.7 and works as follows. The input data set is decomposed over the available processors using “standard” domain decomposition; each processor \( p_{0 \leq i < P} \) reads \( \lceil N/P \rceil \) slices (where \( N \) is the number of slices in the input data, \( P \) the number of processors and \( P \leq N \)), unless \( N \mod P \neq 0 \) in which case processors \( p_{i < N \mod P} \) read \( \lceil N/P \rceil + 1 \) slices. Note that we assume with this decomposition method that the workload for each processor will be the same when they are given equal shares of slices. As this assumption depends largely on the data contents and the threshold value selected for the isosurface, the decomposition method used here will not always result in equal workloads.

![Visualization pipeline used for parallel isosurface modeling.](image)

Each processor proceeds by modeling an isosurface for its local domain using a marching cubes contour filter [151]. The number of primitives in the resulting triangle representation is then reduced using a decimation filter and concatenated to form a complete isosurface [210]. If the concatenated isosurface produced at this point would be used for rendering by OpenGL, an unfortunate side effect of the distribution over multiple processes can result, as illustrated in Figure 2.8. If the local isosurfaces at the boundaries of neighbouring local domains are curved differently, the isosurface patches will be shaded incorrectly. The shading method used in OpenGL is Gouraud
Figure 2.8: Parallel isosurface extraction; domain decomposition (top left); for each subdomain, an isosurface is extracted and the resulting triangle mesh is decimated (top right); lighting artefacts appear on curved domain boundaries caused by ill-defined normal vectors (bottom left); relaxation of the triangle mesh corrects these artefacts (bottom right). The two bottom images have been colour-enhanced to show detail.

shading which requires that a normal vector is defined for each vertex in a surface patch [89]. The normal vector in each vertex is determined by first determining all faces that share the vertex. The normals of all adjacent faces are then averaged to get the vertex normal. At local domains, the normals of adjacent faces on neighbouring domains are not available and the vertex normal is averaged over locally adjacent faces only. The result is an ill-defined normal vector which is most conspicuous on
Design considerations for interactive exploration environments

strong curved boundaries. In our implementation, this problem is solved through a relaxation filter on the concatenated isosurface, implemented as an additional decimation step that is set to only merge co-planar surface patches and recalculate vertex normals. The resulting isosurface will have correctly defined vertex normals and, at the same time, the number of triangles is reduced even further. Finally, a triangle strip filter is used to convert the isosurface into a representation that requires less space to store and that can be rendered at higher efficiency by OpenGL [261]. The end result is transferred to VRE for rendering.

Interaction methods

Interaction in VRE is done primarily with the wand, including its buttons and joystick. In addition, a simple but effective menu system has been built that provides access to all of VRE's functionality. Options on the menu are selected by pointing at one of the items and clicking a wand button. A virtual pointer is rendered from the front of the wand to provide visual feedback; seeing the pointer intersect with the menu as well as the highlighting of the selected menu items aids in the menu selection process. Again, the virtual pointer also helps in unambiguously identifying interesting structures to other users as described in section 1.5.2 (page 19). The menu provides access to the patient data sets stored in the PACS database, enables/disables visualization options and allows various visualization options to be set interactively, including object scaling, rotation, translation, the desired isosurface modeling threshold, the sampling rate used for volume rendering and storing/retrieving transfer tables.

The joystick on the wand is used in two modes, as selected from the menu. In the first mode, the wand is used as in most CAVE applications; it moves the position of the CAVE towards the direction where the front of the wand is pointing, at a velocity proportional to the amount the joystick is pressed forward or backward. Sideways pressure on the joystick is used to rotate around the center of the CAVE, around the y-axis (which in the CAVE points upwards) (see also Figure 2.2, page 28). Although this so-called “navigation” takes some getting used to at first, this method of navigation quickly becomes “natural”. In the second mode, the wand is used to transform the visualized objects: it can be used to scale, translate or rotate the object as specified by a selected transformation in a menu option. For purposes such as virtual endoscopy, the user can use the scaling transformation to “blow up” isosurfaces of structures until they are large enough to inspect the insides of the structure. This scaling is mandatory for this feature: although it is possible to view the insides of rendered object merely by “sticking ones head in”, this can be a most unpleasant strain on the eyes as they have to accommodate on structures that are too close [99]. Note that volume rendering is not suitable for this type of interaction due to the nature of the direct rendering algorithm; the resulting images show up as big blobs of pixels from which no structure can be discerned. Also, on most of today's hardware, scaling up the volume would quickly reach the fill rate limitations of the graphics hardware resulting in low frame rates and sluggish responsiveness.
Clipping is the VRE equivalent of the cine-loop capability used on medical workstations. It allows the user to cut away sections of a visual object (isosurface, volume or both individually) using a plane that reveals the inner parts of a dataset that would otherwise be obscured. Once activated via the menu, the clipping plane is attached orthogonally to the forward-pointing wand, at a constant distance, and follows the position and orientation of the wand. Interaction is very intuitive: the user only needs to move the wand into the visual representation of the scan to inspect its interior. Clipping is implemented using OpenGL’s standard clipping mechanism and is fully hardware accelerated.

![Clipping Plane](image)

**Figure 2.9:** Transfer table editor used in VRE. Scalar values on the horizontal axes are mapped to colour properties on the vertical axes. The editor supports interactive definition of separate mappings for red (R), green (G), blue (B), opacity (OPAC) and luminance-alpha (L-A). Shown here is a luminance-alpha mapping with one additional control point.

The interactive definition of the transfer tables for volume rendering is done using the editor shown in Figure 2.9. The rectangular buttons on the left allow separate transfer tables to be defined for red, green, blue, opacity (or alpha) and luminance-alpha. Luminance-alpha can be regarded as a mapping where red, green, blue and opacity values are identical, resulting in dark/transparent colours for low scalar values to luminous/opaque colours for high scalar values. The buttons marked with arrows on the right and at the bottom allow the transfer tables to be shift and scaled. By default, a linear "luminous-alpha" mapping is defined from dark/transparent for low values (shown by a square in the lower left corner) to luminous/opaque for high values (shown by a square in the upper right corner). The linear mapping between these values is shown by a diagonal line between the two. This mapping can be altered by introducing new control points anywhere on the diagonal line and moving the control points over the window.

### 2.3.6 Experiences

We organized two proof-of-concept demonstrations for a number of radiologists and physicians from the hospital; one took place in the CAVE at SARA, Amsterdam, the other in the radiology department of the Leiden University Medical Center (LUMC,
Design considerations for interactive exploration environments

Figure 2.10: Surface rendering of the abdominal aorta. GEOPROVE (see section 4.5) is used to measure the angle of the bifurcation (see also colour reproduction on the back cover).

Figure 2.11: Interactive oblique clipping of a volume rendering of a human head MRI scan (image created by Anton Koning, SARA).

Leiden). For the latter we installed an IDesk projection system in the department with a Silicon Graphics Octane as the computing and graphics hardware. The data sets used in the demonstrations included patient scans provided by the radiologists as well as scans obtained from other sources, such as the Visible Human data sets [2,174,223]. The VRE application showed how patient data stored in a database on the IBM SP2 at SARA could be loaded via a network connection and visualized using volume and surface rendering techniques (see Figures 2.10 and 2.11). In addition, the system demonstrated how isosurface modeling could be executed on the IBM SP2, the results of which would then be transferred back to the visualization front-end for rendering.

During an evaluation meeting it became clear that the radiologists and physicians regarded the VRE environment as a useful instrument for educational, demonstration and communication purposes. However, they agreed that the quality of the visual representations are insufficient for diagnostic purposes unless applied to very specific situations in which the visual analysis of three-dimensional structures would be required. The foremost problem was the lack of texture and detail in the visual constructs generated by VRE. The most important reason for this is a hardware limitation of the graphics pipelines that were used, both in the Onyx2 of the CAVE and in the Octane; most of the data sets used for the demonstrations had to be downsampled and reduced in scalar resolution to fit in the available texture memory. Furthermore, some of the interaction mechanism provided were regarded as difficult to use, most specifically the definition of colour transfer tables in volume rendering (which in some cases could be solved by providing presets), determining proper isosurface levels for surface rendering (which in some cases could be solved by providing image
2.3 The Virtual Radiology Explorer

Figure 2.12: Execution time (in seconds), speedup and efficiency of the parallel isosurface engine on a high resolution version of the data set shown in Figure 2.8.

histograms) and the techniques for the manipulation of the presented constructs (i.e. rotation, scaling, translation versus navigation in the VE).

In addition to this, there are also practical reasons why radiologists will not adopt this kind of technology right away. Radiology departments do not have the funds, manpower or space to house and maintain a CAVE installation. Nor will they be willing to spend time to travel to a CAVE installation elsewhere. Smaller systems like the IDesk come a little closer to a solution as these can be used in the radiology department. Still, these systems are too expensive. In Chapter 3 (page 51) we describe the construction of low-cost VR systems based on off-the-shelf computing hardware.

An addition to the VRE system that would make the system more useful was found to be a capability that allows measurements to be taken from the visual presentations. Physicians often need information on the size of certain anatomy and/or pathology in order to prepare a surgical procedure. Obtaining measurements from visual representations in a VE is an area in which very little research has been done. We took up that challenge and present an architecture to do measurements in VEs, called GEOPROVE, in section 4.5 (page 86). Figure 2.12 shows the performance characteristics of the parallel isosurface engine while modeling the isosurface of the skin from a CT scan data set of $256 \times 256$ pixels per slice, 94 slices, 16 bits per pixel (this is a high resolution version of the data set shown in Figure 2.8 and with the same threshold value). This figure shows that the extraction of isosurfaces on multiple processors does result in a reduced execution time but that the efficiency of the parallelization is far from optimal. As already noted
earlier, this is caused by the domain decomposition method that was used. Indeed, performance measurements of the same algorithm on uniform data, resulting in an equal workload on all subdomains, show an almost linear speedup (data not shown). To increase efficiency of the program, methods should be introduced to balance the workload over the available processors. Because the performance of the current implementation was considered acceptable for our purposes we have not pursued these methods. Various methods could be used to improve the workload balance to obtain better efficiency such as alternative data decomposition methods or through farming of the data in small portions over the available processors.

2.4 Diffusion and flow limited biological growth

In our research group there is a strong interest in the study of biological systems [111–121]. Computational simulation models play a fundamental role in the study into the behaviour of these systems. What is frequently missing are methods to analyse the results of these simulation models that help in their validation. Frequently, automated analysis of the simulation results is difficult due to the unavailability of suitable algorithms or, in cases where algorithms do exist, the computational requirements are too high to perform an effective analysis. In the following case study, we address a computational model for the study of marine sessile organisms, such as sponges and stony corals. We describe an immersive visualization environment that is used for the interactive visual analysis of simulation results.

2.4.1 Background

In the development of many biological systems, the distribution of chemical agents and nutrients plays a fundamental role. For filter-feeding marine sessile organisms, such as stony-corals, the growth process is affected by the distribution of suspended material in the external environment. From the biological literature it is well-known that water movement may have a strong impact on the shape of stony-corals. It is often possible to correlate growth forms of stony-corals with the amount of water movement. Compact growth forms are generally found under conditions with a large exposure to water movement, while the growth form changes gradually into a branching shape when the amount of water movement decreases. Figures 2.13 and 2.14 show two growth forms of the stony-coral species Pocillopora damicornis. The compact form in Figure 2.13 originates from an exposed site and the thin-branching form in Figure 2.14 was collected from a sheltered site. 

Our research group has studied the effect of hydrodynamics on a very simple type of growth process, viz. growth by aggregation. In this model, aggregation proceeds by the accumulation of a "nutrient". The nutrient distribution is modeled using a Lattice Boltzmann model of transport. The aggregate absorbs the nutrient and the amount absorbed determines the local growth probability. We have carried out simulations of growth processes (aggregation processes) in which an aggregate consumes nutrients
from its environment and where nutrients are dispersed by a combined process of flow and diffusion [121]. The effect on the aggregate caused by different rates of fluid flow and nutrient dispersion is investigated.

2.4.2 Analysis of simulation results

The data resulting from these simulations includes the growth of the aggregate over time, the dispersion of nutrients around the aggregate, the absorption of nutrients on the surface of the aggregate and the velocity of flow around the aggregate. Growth in the data is encoded as a three dimensional volume of grid nodes $V(x,y,z)$ where $V(x,y,z) = t$ denotes that the grid node at $(x,y,z)$ was aggregated at time step $t$ (and $V(x,y,z) = 0$ in grid nodes where no aggregation took place). Originally, the only method to obtain insight in the results of the simulation was through visualization of the generated data sets and visually comparing these with existing coral structures. A special purpose software package was implemented to obtain surface models of the generated structures at each growth step $T$ by extracting an interpolated isosurface from the volume through all grid nodes where $V(x,y,z) = T$, which were then visualized on a graphical workstation. This resulted in a 3D surface representing the shape of the aggregate. However, the complexity of the aggregates was such that the generated surface models were too big to allow interactive exploration. In such cases it
was often necessary to generate animations on video which took well over a week to produce. In addition, the end results were inherently non-interactive which impedes exhaustive exploration.

### 2.4.3 Interactive exploration in Virtual Reality

To allow the simulation data to be explored interactively, we have built an environment that allows the simulation data sets to be explored inside a CAVE. This environment supports interactive visualization of surface models of the aggregate at any time step, animated playback of the development of the aggregate from start to finish, surface models of the nutrient distribution around the aggregate and colouring of the aggregates based on absorption.

For our initial experiments towards the development of an exploration environment we have taken a simulation model for the investigation of diffusion and flow limited biological growth. We have used data sets resulting from simulations of growth processes (aggregation processes) in which an aggregate consumes nutrients from its environment and where nutrients are dispersed by a combined process of flow and diffusion. Details about the simulation model are given elsewhere [121]. As an example, the effect on the aggregate caused by different rates of fluid flow and nutrient dispersion is investigated. The data resulting from these simulations includes the growth of the aggregate over time and the dispersion of nutrients around the aggregate. This model has been used as a simple model for coral growth [121].

Our interest is to compare the simulated structures with structures found in nature, the investigation of the complex geometry generated by the simulation and the flow fields around it, the behaviour of tracer particles released in the flow field, the location of nutrient absorption points, and the location of pressure fields causing hydrodynamical forces on obstacles. A crucial issue in the development of morphological simulation of growth processes is the ability to compare simulated growth forms with the actual objects. For this reason we have compared data sets of actual objects to the simulated growth. The data on the actual objects were obtained from CT scans made of some samples of the stony-coral *Pocillopora damicornis*. These CT scans consist of slices that each contain 512 by 512 16-bit gray scale values, where the number of slices depends on the length of the object and the number of rotations made by the scanner. The model shown in Figure 2.15 was reconstructed from 30 CT slices from which a surface contour was generated using an isosurface extraction algorithm in the Visualization Toolkit [209]. The CT scanner used for this scan was a Philips Tomoscan SR7000. The relatively low number of slices results in a decreased resolution in one of the principal axes which poses some problems in the reconstruction of a model that should be accurate enough for quantitative comparison.

Although our exploration environment provides methods by which the data sets can be explored visually, an important aspect in the development of any simulation model is its verification against the system that is modeled. A major problem in the quantitative comparison of the simulation results with actual phenomena is that in many cases there is no single discriminative feature by which they can be differentiated.
In our test-case for example, a property such as the fractal dimension \[155\] gives some insight in the global resemblance of different structures but is inadequate in describing the quality of the simulation model as only a limited aspect of the overall morphology of an object is captured. Therefore, it is often more suitable to obtain measurements on multiple properties in local areas of the data sets that together form a discriminative measure.

In case of the growth model we wish to compare the shape of the resulting structures to those found in nature. In previous work it has been demonstrated that morphological properties such as for example the thickness of branches and the shortest distance between neighbouring branch points ("branch spacing") provides relevant biological information and can be used to compare simulated with actual growth forms \[118,119\].

In addition, when comparing simulated coral objects with actual corals, the comparison procedure should be non-destructive to the real coral as most of these are valuable and irreplaceable specimens. However, this makes many measurements difficult if not impossible since the complex shape of these structures prohibits the use of instruments that may damage the coral. One possible solution is to acquire a sufficiently accurate three-dimensional scan of the coral. Since conventional photographic or laser scanning techniques are only suitable for obtaining surface models of objects which have no obstructing components, these devices are unsuitable for scanning complex and irregularly structured objects such as corals. Fortunately, we have obtained digital 3D data sets of a number of corals which we can use for our purposes through the assistance of the Radiology department of Leiden Academic Hospital, who have graciously offered to scan the corals with a computed tomography (CT) scanner.

Although the properties we want to measure could be obtained automatically using
data analysis techniques, this often requires designing and implementing specialized algorithms that are dedicated to the specific task. Quite often these techniques rely on heuristic algorithms that are difficult to design, implement and control. An interactive environment equipped with a system that allows measurements to be taken from the visualizations that are rendered in the virtual environment can provide the techniques needed to acquire quantitative properties from data sets which would have been difficult to obtain otherwise. We have designed and implemented a system to support this, called GEOPROVE which will be described in detail in section 4.5 (page 86).

### 2.4.4 Exploration in the CAVE

We have built an interactive exploration system that allows the data sets, generated by the simulation, to be explored inside the CAVE located at SARA. Within this interactive exploration system, surface models were used for visualizing the growth of the objects, a tracer distribution model for studying tracer distributions about complex objects and for measuring the degree of absorption of tracers at the objects, methods for visualizing the flow field around the objects, and methods for sectioning the objects which enables us to study the addition of material during the growth process. Figure 2.16 shows the result of a simulated aggregation process. The aggregate emerges in an environment where nutrients are mainly dispersed by diffusion. In
this case an irregular branching aggregate is formed. The colour of the object represents age; from dark red for “old” parts, to white for “young parts”. The nutrient distribution around the aggregate is visualized using a blue-white gradient, where blue indicates a high and white a low nutrient concentration. The exploration environment facilitates the interactive inspection of these gradient planes by allowing the plane to be moved through the simulated structure over all three principal axes.

2.4.5 Discussion

Using the developed exploration environment we have investigated simulated results of various experiments in which the influence of hydrodynamics on the growth process was varied. In addition, we have been able to compare these results with CT scans of actual stony-corals. Using these CT scans, a more flexible comparison of the simulated structures to those found in nature is now possible. The qualitative comparison showed that both the simulated growth forms and the actual stony-corals show a similar tendency: when the influence of hydrodynamics increases, both simulated and actual forms exhibit an increase in compactness. This observation corresponds to the observations reported in [121]. The exploration system has also shown differences which were not detected before: when comparing the CT scans to the simulated results it was found that there is a difference in the branching patterns. Especially in the compact aggregates branches tend to fuse (anastomosis) easily, while this phenomenon was not detected in the CT scans. This observation indicates a difference between the actual growth and the simulation model.

Our main finding is that the use of this exploration environments in the CAVE allows us to study the effect of flow on the nutrient distribution far more easily than was previously possible. The environment allows us to study the morphology of the aggregates simply by walking around the presented object. Using the CAVE’s “wand” we are able to explore the growth of the aggregate over time, and the dispersion of nutrients around the aggregate.

2.5 Summary and conclusions

In this chapter we described three interactive exploration environments: one for the exploration of car crash simulation results, one for the exploration of 3D medical data sets and one for the exploration of simulation results of a diffusion and flow limited biological growth modeling simulation. Our aim was to obtain a better understanding of the different issues involved in the design, implementation and use of these interactive exploration environments.

To obtain usable environments in terms of the requirements described at the start of this chapter (i.e. quality of presentation, rapid frame rate, intuitive interaction and real-time feedback), we had to make compromises. Some of these have resulted in minor nuisances. For example; in the car crash environment, the use of display lists increases application startup time but results in higher frame rates and increased
responsiveness. Also, the parallel isosurface modelling method used in the VRE environment results in models that can be displayed at high frame rates but at the cost of increased response time. Other compromises are far more serious; the reduced image quality as a result of both the down-sampling and the reduction of scalar resolution of medical data sets (to compensate for hardware limitations), results in visual representations that medical experts find insufficient for diagnostic purposes. Also, in both the VRE and the biological growth environment, it is frequently impossible to construct an acceptable quality isosurface model that, at the same time, consists of sufficiently few triangles that they can be drawn at high frame rates. As a result, users are often confronted with high quality isosurface models with which interaction is almost impossible due to low frame rates, or with low quality models that can be displayed at acceptable frame rates. Most of these problems are the direct result of graphics and computing hardware limitations. Given time, the increase in performance and capabilities of future hardware may allow at least some of these problems to be overcome. However, as already noted in chapter 1 of this thesis, an increase in computing performance is invariably accompanied with an increase in problem size so that hardware limitations will always remain a problem.

Of all the experiences our users had with the environments, what had the most impact was the acuteness of the immersive experience in the virtual environments. Almost instantly, the users of our environments were impressed with the fidelity of the virtual objects floating in front them, which in some cases even made them want to reach out and touch the virtual objects. This quality can be mostly attributed to the combination of the surround-screen projection system of the CAVE, the stereoscopic images and head motion parallax. Our environments have also been used in other research projects, which in some cases resulted in the discovery of artifacts that had not been noticed before. For example, the VRE environment has been used for the visualization of 3D confocal laser scanning microscopy (CLSM) data from the molecular cell biology research group of the University of Amsterdam where they found spatial configurations in the nucleus of biological cells that had not been detected before.

The biggest problem for most inexperienced users was the interaction with the environments. In relatively simple environments (such as the car crash simulation playback environment), the number of functions offered by the application are sufficiently few that users have little difficulty interacting with the environment. However, in more complex environments, it has proven to be difficult to provide the environment's functionality in an intuitive manner. For one, this can be attributed to the lack of graphical user interface (GUI) toolkits for virtual environments. These toolkits would enable the application developer to extend the interaction through hardware devices (like the wand, buttons and joystick) to graphical interaction techniques (such as graphical buttons, sliders and menus) to create user interfaces that the user knows well from desktop applications. In chapter 4 we describe several techniques that address interaction in virtual environments.

All the environments described in this chapter have in common that the explored data is static or time-invariant; the data has been collected at a specific point in time; after data acquisition or after a simulation has finished. Chapter 5 addresses the
issues involved in the construction of dynamic environments that allow a researcher to explore the progress of computational processes while they are running.
Design considerations for interactive exploration environments