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: http://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 1

Interactive exploration environments

"Computers are useless. They can only give you answers."

Pablo Picasso (1881-1973).

1.1 Introduction

The increase in availability of computational resources, both storage capacity and processing speed, have allowed researchers in industrial and scientific areas to investigate increasingly large and complex time dependent problems. As a result, the data sets that are generated by these applications grow larger and more complex. Furthermore, many of the industrial and scientific applications are typically simulations of complex systems. In general these problems are intractable and NP complete so that the only available option to obtain insight in these problems is through explicit simulation. As in this class of problems the parameter space is typically extremely large, it is not feasible to simulate every point in the parameter space. Optimization algorithms that perform a guided search through the problem’s parameter space have been used to avoid this [221], still, as these parameter spaces grow larger, the time that is required to find a satisfactory solution is unacceptable.

In many cases, the automated analysis of these data spaces is difficult, either because no techniques are available to extract the features of interest or otherwise simply because it is unknown what information is present in the data beforehand. Interactive exploration may be one of the few options to analyse this data in order to obtain further insight in these data spaces.

1.2 Interactive exploration

Often, the only alternative to obtain insight in large, complex data sets in cases where the automatic analysis of data spaces is impossible is through human inspection. Here, a human being takes on the role of analyst who uses his knowledge to analyse
the data. Unfortunately, the increase in computing capacity is not paralleled by an equivalent increase in human capacity. Hardly ever will the analyst resolve to inspection methods where he scrutinizes piles of numbers in search of important clues since this is cumbersome, prone to subjective errors and not in the least, mind numbing. Instead, the data is converted into a representation that allows the analyst to “explore” the data and perceive patterns that will help him find structure. If performed correctly, the analyst’s senses, cognitive abilities and experience together will help him in forming a mental picture that leads to a better understanding of the data. The primary goal of exploration therefore is to increase the bandwidth to the brain.

The challenges in creating an environment that is suitable for this type of exploration are many. First, the environment should be able to represent data in a correct, clear and informative manner. The conversion of data into another representation leads to the risk of introducing patterns in data that really are not there, so care must be taken in which methods are used. Indeed, it is often a good idea to provide multiple methods so that the user has the ability to interactively choose between different representations of the same data. Second; the conversion of data into another representation inevitably takes time. The amount of time should be minimized in order to obtain an environment that is responsive to user interaction. Third; exploration environments will only be effective if the explorer does not feel restricted by the environment during his work. The user should be allowed to interact with the different aspects of his experiment in an intuitive manner. In addition, the environment should provide clear and rapid feedback in response to user interaction.

Unfortunately, we will see that these different problems are hard to solve all at once. As each design choice has its implications on the performance of another we will see that compromises are often necessary. In this work we distinguish two types of environments; static and dynamic exploration environments.

1.2.1 Static environments

In Interactive Static Exploration Environments (ISEE), the data presented to the user is time invariant. The data that is to be explored is generated by an external process, such as a computer simulation or a data acquisition device. Once the data is loaded into the environment, the user is presented with a representation of this data and provided with interaction methods to change the representation parameters interactively to obtain the best “view” required to gain more understanding. Each change of a parameter results in an update of the presentation. The data itself, however, does not change (see Figure 1.1).

Example areas where ISEEs are applied are plentiful. They can be found in medicine for the visualization of data obtained from medical scanners such as Computed Tomography (CT) and Magnetic Resonance Imaging (MRI) [41,166,274]. Here a radiologist decides on the proper settings of a visualization environment for the visualization of a medical scan that aids a surgeon in assessing a diagnosis for a particular patient. In molecular biology, visual exploration environments are used to obtain a
better understanding of the spatial structure of complex molecules [5, 92, 144]. In architecture, these environments can help in assessing the aesthetics of a building before it is built [150].

**Data presentation and interaction**

An important step towards a successful ISEE is to involve the researcher into the presentation as much as possible, thereby increasing the researcher's level of awareness [29]. To achieve this, an exploration system needs the following, mostly interdependent capabilities:

- **Informative presentation** – The most commonly used method to provide an insightful representation of complex data sets is by visualization. Here the abstract data are rendered as visual constructs that represent quantitative and relational aspects to the observer in an intuitive manner. This is the area of scientific visualization [27, 157, 170, 209] and information visualization [39, 222, 238]. Many visualization environments are available that provide means to efficiently achieve this [103, 106, 172, 242]. Although visualization is a very powerful and well understood method for data representation, other sensory modalities such as sound, touch and even smell, or combinations of different modalities may in some cases lead to better results.

- **Persistence** – The time that is needed to generate a rendering from start to finish can often not be easily dismissed. In the case of visual renderings, persistence (the rate at which consecutive frames are perceived as continuous motion) is obtained when at least 25 frames are rendered per second. Sound waves are perceived as tones from around 20 cycles per second and up. Humans can feel temporal frequencies starting from around 1 kHz and therefore rendering rates for haptic displays must be in the order of kilohertz. Update time is often dependent on the “level of detail” (LOD) in the presentation. In these cases a balance between the LOD in a presentation and the maximum allowed update time is necessary (the aim should be to employ “minimal means for maximum effect”).

- **Intuitive interaction** – A prerequisite for effective exploration is that a sufficiently rich set of interaction methods is provided that allows a user to modify the parameters that control the presentation in order to be able to extract both
Interactive exploration environments

provide qualitative and quantitative information from the underlying data sets. An unfortunate side effect of increasingly richer sets of interactive methods is that user-friendliness is often compromised, so careful consideration is required during user-interface design. The ground rule for interaction capabilities is that interaction should be as intuitive as possible: intuitive interaction methods should require no explanation [171,204].

- Rapid response - Some delay will always occur between the moment a user interacts with a presentation and the moment that the results are visible. This could be caused by many factors, including low tracking rates of input devices, communication delays and temporary reduced availability of computational or network resources. To attain accurate control over the environment and to avoid confusion with the user, the amount of lag should be minimized. In general, real-time interaction encourages exploration [29].

Provided these capabilities are carefully considered, these environments are well suited for the exploration of static multi dimensional data sets [13]. The design issues in the construction of an ISEE will be described in more detail in Chapter 2.

1.2.2 Dynamic environments

Interactive Dynamic Exploration Environments (IDEE) extend the previously described static model in that the information provided to the user is regenerated periodically. Again, the data originates from an external process which in this case is an active member of the exploration environment. The external process repeatedly generates new data, either autonomously (without user intervention), on-demand (as a result of user intervention) or both. In addition to the capabilities described for the static environment, the dynamic environment is expected to provide (1) a reliable and consistent representation of the results of the external process at that moment and (2) interaction mechanisms that enable the user to change parameters of the external process and of the presentation. Again, each change of a parameter in the external process or the presentation results in an update of the environment. New data generated by the external process results in an update of the presentation (see Figure 1.2).

While interaction in an ISEE influences the presentation only, in dynamic environments interaction influences both the presentation and the external process. On an implementational level this requires additional processing code to service interaction. On an operational level this change may influence the prerequisites described in this and the previous subsection. These design issues will be described in more detail in Chapter 5.

Interactive systems such as the ones described here allow for live experimentation by a researcher. These systems are called by many names; computational steering environments (CSE) [178,245], user-steered calculation [100], problem solving environments (PSE) [84], human-in-the-loop (HITL) [235] computing, virtual laborato-
1.3 Scientific visualization

The key method to present data for the purpose of exploration in use today is visualization. The main reason for this is that the human visual system is thought to be the most important of the human senses; it has high bandwidth and allows natural communication as the human visual sensory system is capable of understanding complex shaped image renderings with relative ease. When visualization is applied to the representation of scientific data we speak of scientific visualization. A report of the National Science Foundation (NSF) in 1987 described scientific visualization as “the integration of computer graphics, image processing and vision, computer-aided design, signal processing, and user interface studies” [157]. As such, scientific visualization would not only entail the visual representation of scientific data but also image synthesis using computer graphics techniques, data analysis, engineering and human perception. This definition is too broad for our purposes. Here we will limit ourselves to a narrower interpretation in a scientific computing context, viz. the generation of visual representations from the results of scientific applications.

The basic objective of scientific visualization is to create a mapping of data structures to geometric primitives that can be rendered using computer graphics techniques (often points, lines, triangles, squares or polygons). Although this basic principle is easy enough, the real challenge in scientific visualization is to create a mapping that creates a representation that is understandable, correct and complete. Scientific visualization is applied in many different research and industrial areas. Often these applications involve the visualization of processes that are difficult to observe directly by the human visual and/or cognitive system. Phenomena of interest to the scientific and industrial community often include topics that can not, or not easily be studied directly because they are too small (molecules, atoms, etc.), too big (planetary systems, galaxies, etc.), too dangerous (nuclear explosions, combat situations), too slow
(colliding galaxies, stock rates), too quick (quantum processes), too concealed (organs in the human body), etc. Instead, methods are used to acquire and record data from the event of interest, the results of which are then visualized. In all cases it is important to realize that the visualization of these applications provide a representation of the underlying real world phenomena. This should always be kept in mind while interpreting the visualization as it could lead to misinterpretation.

### 1.3.1 Display techniques

In its earliest form, the visualization of scientific data was performed on systems that produced 2D vector representations. With the introduction of powerful raster displays and new algorithms, more complex graphics techniques came into reach of the scientist which included the calculation of 2D projected rasterizations of 3D scenes [79] and of lighting models such as smooth shading of polygonal surfaces [89, 183]. The dramatic increase in performance of graphics hardware over the last years allows these methods to be used for the rendering of complex 3D scenes in real-time. Thanks to the personal computer (PC) gaming industry, the cost of this graphics hardware has dropped dramatically as well, allowing scientists to create advanced scientific visualizations on cheap and commonly available hardware.

![Simplified representation of a rendering pipeline.](image)

A useful way to describe the method to create raster scanned images from 3D scenes is through a rendering pipeline [79]. A simplified form of a rendering pipeline is shown in Figure 1.3. The rendering pipeline is split into two parts: the **geometry processing** and the **pixel processing** phase. In the geometry processing phase, each primitive in the scene is traversed and transformed (translated, scaled, rotated) from model coordinates to world coordinates using a linear transformation matrix. Next, a viewing transformation is applied based on the location of the viewer and all primitives outside the view frustum are clipped. In the pixel processing phase, the clipped primitives are scanned into screen pixels (including lighting, shading, depth calculation and texture mapping). Finally, the resulting image is stored and composited (including alpha-blending and depth-buffering) and converted to an analog video signal by a digital-to-analog converter (DAC) for display on a monitor or projector.

Later and still ongoing research has resulted in ways to increase the performance of the graphics pipeline through multiprocessing and dedicated hardware techniques [4]. A major player in the success of computer generated 3D graphics in this respect was Silicon Graphics, Inc. (SGI), founded by Jim Clark in 1982 after having spent four years at Stanford University with the expressed purpose of developing hardware...
1.3 Scientific visualization

technology that was called a “geometry engine”. The geometry engine contained functionality that was able to almost instantaneously perform the complex geometrical mathematics required for 3D graphics that would otherwise have required thousands of lines of program code. It was not until 1985 that SGI put its first workstation on the market, but from that time on they continued to market systems with a graphical performance that was unsurpassed by other companies for many years to come. SGI’s encounter with the Hollywood film industry in the early 1990s resulted in a huge exposure of Computer Generated Imagery (CGI), amongst which were blockbusters like Terminator II (1991), Jurassic Park (1993) and Toy Story (1995).

1.3.2 The visualization pipeline

The accepted approach to visualize data structures is through an extension of the graphics pipeline described in section 1.3.1. This “visualization pipeline” is, as the name suggests, a first-in-first-out (FIFO) structure where each stage accepts data at its input as soon as it is presented, transforms the data and provides its output to the next stage (see Figure 1.4). The “source” at the start of the pipeline reads or generates the data that is to be visualized. This data is transformed by one or more “filters” into a representation that is suitable for graphical rendering. A “mapper” transforms this representation into geometric primitives that can be drawn by the rendering engine. Finally, at the end of the pipeline the renderer takes the geometric primitives for transformation into pixelated images.

![Visualization Pipeline Diagram](image)

*Figure 1.4: Simplified representation of a visualization pipeline.*

Although some stages may be able to come up with reasonable defaults to perform their function, most stages in a visualization pipeline have to be configured in order to produce the results the user is looking for. To do so, each stage is configured using a set of parameters. A modification of parameters in a stage requires that at least this stage and all stages “downstream” are updated. To achieve this, the visualization environment checks the pipeline in the direction opposite to the data flow direction and instructs the first modified stage that it encounters to update itself. Once the modified stage is updated it produces new data, thus forcing subsequent stages to execute as well. In doing so, only part of the pipeline needs to execute which reduces computational overhead while maintaining consistency.

One example of a visualization environment that uses the pipeline architecture described here is the Visualization Toolkit (Vtk) [106,209]. Vtk is an open source, freely
available collection of classes with contributions by a lively user group from all over the world. The types of functions in Vtk can be classified into graphics, image processing and visualization. The Vtk visualization pipeline is similar to that shown in 1.4 and will be described in more detail in section 4.2.

1.3.3 Interactive Scientific Visualization

The performance increase in computing and graphics hardware allow the long-term wish of many scientists to closely interact with their models. The NSF Scientific Visualization report envisioned a situation where large scale computations would be carried out on high performance computers while rendering and interaction would take place on a visualization workstation. The two would be connected by high-speed networks to cope with the high volumes of data that would have to be transferred. It would allow a researcher to connect to a running application, inspect its data structures and change values in order to understand its behaviour. This prospect would support a scenario that was considered not only possible but essential in the scientific discovery process.

With the increase in performance of computers, the possibility for researchers to perform larger and more complex simulations increases as well and, as a result, the demands posed on visualization workstations. To enable interactive exploration, the workstations would have to be able to do interactive rendering. At the same time, the complexity of the simulation models requires advanced visualization methods in order to represent the models and the simulation results in a comprehensive manner. This added functionality will reflect itself in additional complexity regarding the use of these environments, so improved user interfaces would be required to permit the operator to interact with the applications.

1.4 Virtual Reality

As noted in the previous sections, one way to support the scientific discovery process is by allowing the researcher to actively explore the processes that take place in his models. By increasing his involvement, the researcher would be able to gain a better understanding of the underlying models. A way to achieve this is to represent the data or processes in such a way that they can be “experienced” using the human senses. In general, as more of the human senses are stimulated by the events that take place around them, the more involved they get with the experience. The promise made by advocates of Virtual Reality is that computer technology facilitates the construction of devices that provide sensory stimuli to as many of our senses as possible in order to create the ultimate reality experience.
1.4 Virtual Reality

1.4.1 Historical background

There is no exact definition of what Virtual Reality (VR) precisely is. This is in part because of its historical association with a colourful diversity of social cultures, varying from game developers [141], arts movements [110], lyricists [11], visionaries that saw VR as the alternative drug [143] and other popularity writers [203]. Advocates have for some time even been reluctant to associate themselves with the term and have used alternatives such as “Artificial Reality” [137], “Cyberspace” [87] and later the most commonly used “Virtual Worlds” and “Virtual Environments”. In this thesis we will use the name Virtual Reality (VR) for systems that are capable of generating interactive artificial worlds. We will use the term Virtual Environment (VE) for the environments created by VR devices.

Man has always looked for ways to escape reality and engage into other, more fantastic experiences. Some feel the origins of VR go way back to the time when primitive man painted pictures on the walls of caves [195]. Since the beginning of the written word, book and story-writers have recorded the figment of their imagination for the enjoyment by others. Playwrights left less to the imagination of their audience by enacting stories on stage with the sole purpose to provoke strong emotional responses. With the invention of the moving picture and television, cinematographers and television program makers captured the imagination of millions. Their stories became even more realistic over the years with the introduction of new technology that made the experience more compelling and breathtaking, such as the introduction of (surround) sound, larger screens, special effects and stereo projection techniques. However, when we compare them to the real world, all these alternative worlds have two major shortcomings before we may truly call them “reality experiences”. First; we experience the world around us through our senses - our eyes, ears, nose, skin and tongue - yet, just a few of these are used to persuade us into believing we are “elsewhere”. Second; few of them allow the viewer to influence the sequence of events that have been preprogrammed by the storytellers. They are all, essentially, non-interactive.

Virtual Reality pioneers

In 1956 Morton Heilig, a Hollywood based cinematographer, proposed that the next evolutionary phase after the cinema theatre would be an environment where the viewer experiences not only images and sounds but also odors and touch. Heilig believed that by doing so the barrier between the viewer and the theatre would dissolve, creating a total illusion which he called the “experience theatre”. Heilig’s work led to “Sensorama”, a device he designed and patented in 1962 (see Figure 1.5). Sensorama was the first multisensory arcade game where the viewer would sit on a type of motorcycle. As the “passenger” was looking at three dimensional images of the Californian sand dunes through a binocular display, the motorcycle handhelds would tremble in sync with the images while breezes and odors were released from small grilles around the nose and ears. Although Sensorama was called a “game”, the experience was pre-recorded and played back for the user; the experience could not be controlled by the viewer and was therefore not interactive. Sensorama was no cash success but Heilig’s
use of multisensory stimulation testified of great vision nevertheless.

In 1965 Ivan Sutherland at the University of Utah proposed what he called the “Ultimate Display” [230]. This display would enable a person to experience a synthetic computer rendered “Virtual World” as if it were real. In 1968 Sutherland realized a binocular display which he called a “head-mounted display” (HMD) [231]. This device consisted of two cathode-ray tube (CRT) displays mounted on a helmet that projected images into the eyes giving the user a three-dimensional, stereoscopic view of the generated images. The helmet was connected to a contraption, aptly named “Sword of Damocles”, which was suspended from the ceiling and could track the position of the wearer’s head (see Figure 1.6). When the user moved his head, a computer would recompute the images rendered on the displays so that the user would get the impression that the virtual objects were stationary as the user moved around. Sutherland did not pursue the development of wearable displays because the technology available to him at that time was too primitive. Instead, he turned to work on the fundamentals of computer graphics hardware and software design. In 1968 Ivan Sutherland
together with David Evans founded “Evans & Sutherland” (E&S), a company that received high credits in the development of advanced graphics systems that were used in aviation and military simulators. Sutherland received the 1988 ACM Turing Award for his numerous contributions to computer graphics and the 1998 IEEE John von Neumann Medal for “pioneering contributions to computer graphics and micro-electronic design and leadership in the support of computer science and engineering research”.

The work by Sutherland inspired many scientists in different research areas. One of them was Frederick Brooks, Jr. who in 1971 used scientific visualization techniques and a force feedback device for the representation of large molecules [28]. Around that time, VR technology had also slowly progressed to a state where large industries were seeing its potential [150]. During the Apollo missions of the late sixties, NASA used simulators that were used to simulate docking procedures of the Lunar Excursion Module (LEM) and the command module (CM). Another application of VR technology emerged in the aviation industry in the form of flight simulators. Flight simulators are used to train pilots of aircrafts before embarking on their first actual flight. These devices would use graphic displays, sound systems and motion platforms that could realistically reproduce the behaviour of an aircraft during all stages of air travel, from take-off to landing, from normal situations to the most dangerous scenarios.

The defense industry used simulators to train soldiers before missions in “war game” systems that were connected to a distributed virtual environment called SIMNET, linked in real time, involving armored tank simulations [154, 220]. These combat simulations systems were used in the preparation of “Operation Desert Storm”. Other research areas where VR technology is deemed to have great potential is in training and education [158].

1.4.2 A taxonomy of VR systems

Exactly what may be called a VR system remains a topic of debate. Some feel that a VR system should be able to generate an environment that is indistinguishable from the real world. Some even go the lengths of developing an extension of the Turing test to this end [10]. Others say that a VR system should be able to immerse the viewer into an artificial, yet convincingly real environment. These systems would have to obscure the real world and at least have to track the head’s position and orientation. Still others say that a VR system must track the user’s head but may use a desktop display. From a marketing standpoint, just about any system that displays animated 3D graphics seems to merit the VR label. The following types of VR systems can be distinguished (based on a taxonomy by Jerry Isdale [108]):

- Desktop VR - These systems are also sometimes called “Window on World” systems (WoW). In these systems, the conventional desktop monitor is considered as a window onto the virtual world. More realism can be obtained through stereoscopic images, often produced through liquid crystal display (LCD) shutter glasses, which make images “pop out” of the screen. The term “Fish Tank VR” is used when these systems are augmented with a head tracker [250]. The head
tracker is used to change the view on the virtual world based on the position of the wearer. The resulting effect, called “motion parallax”, gives the viewer powerful clues as to the relative distance and size of the objects in the virtual world.

- **Video Mapping** - In these systems a video input signal of the user in the real world is mixed with the virtual world. The user watches a monitor or a projection on a large screen that displays the silhouette of his body interacting with the virtual world. The most famous example of this type of VR was the “Mandala” system [150]. Because the viewer looks at himself interacting with the virtual world on the display, this type of VR is also called “world centered” or “second person” VR.

- **Immersive Systems** - Immersive systems immerse the viewpoint of the user inside the virtual world. To accomplish this they often use display technology that engulf the viewer as much as possible, such as HMD devices or multiple large projection screens, in an effort to overwhelm the viewer with a view on the virtual world and distract him from the real world. HMDs in this respect are sometimes said to provide an “out of body experience” since the user will not be able to see his own physical appearance, or any other physical object in the real world for that matter. As the viewer in an immersive VR system experiences the virtual world almost directly, this type of VR is also called “user centered” or “first person” VR.

- **Telepresence** - In telepresence, the virtual world is not necessarily artificial. Instead, remote sensors, such as video cameras, are used to act as “extension cords”, linked to the human senses to create a on-location view of a real world experience. Examples of where these systems are in use is with firefighters and bomb squads where remote controlled robots are equipped with cameras and robotic actuators to assist during hazardous situations.

- **Augmented Reality (AR)** - In these systems the user’s view of the real world is enhanced or augmented with additional information generated from a computer model. This type of VR is also called “Mixed Reality” or “see-through VR”. Display devices that are used in these systems are called heads-up displays (HUD); they often consist of semi-transparent material that allows the viewer to watch the real world while computer generated images are overlaid providing additional information. Examples are systems used by fighter pilots where the displays provide cockpit information on the inside of the pilot’s helmet.

In this work we focus our discussions on the use of immersive VR systems, such as the SARA CAVE that will be described in section 1.5, although we will in some cases address the use of desktop systems as well, such as the UvA-DRIVE system described in Chapter 3 [14].
1.4.3 Components in a VR system

Any interactive VR system consist of three components [265]. The first is the rendering component which transforms abstract data into a representation that can be perceived by one or more of the human senses. The display component converts these renderings into sensory stimulations for the human perceptive senses. The last is the interaction component which is responsible for determining the position and gestures of the viewer to which the environment should react.

3D computer graphics

Most of today’s high performance graphics workstations contain dedicated hardware components that accelerate the various stages of graphics rendering in the pipeline such as those depicted in Figure 1.3. The hardware architectures vary from vendor to vendor but most use dedicated memories (for textures, depth buffers and frame buffers) and processing units (for transformation and lighting, shading, antialiasing, z-buffering, image composition and texture mapping) that interact in parallel. These dedicated graphics systems relieve the main central processing units (CPU) for doing other tasks. One of the main vendors of computer graphics chips today, nVidia, has coined the term Graphics Processor Unit (GPU) for these devices. The industry standard Application Programming Interface (API) to take advantage of these graphics devices is OpenGL [16, 189, 261]. The OpenGL standard was specified in order to encapsulate device-dependent code and thus to promote program portability to other operating systems. The interaction between the application and OpenGL takes place on the level of polygons, light sources and linear transformations to specify 3D scenes. On a higher level, frameworks such as OpenInventor, World Toolkit and Performer allow the user to specify high-level 3D scenes using object-oriented methods [190, 212, 236]. These frameworks encapsulate many of the more intricate complexities associated with increasing rendering performance (such as scene culling) and interaction with 3D scenes (such as object intersection).

Stereoscopic display technology

VR display technology can be subdivided in two categories: head mounted displays (HMD) and head tracked displays (HTD). The difference between the two is that with HMD systems the display is connected to the viewer’s head, for example in the form of a helmet, so that wherever the viewer’s head moves, the display moves. In HTD systems the display is stationary, as with desktop monitors or projection screens. In both cases the display must portray a stereoscopic pair of images to the viewer so that he sees the virtual objects as if they float in front or behind the display. The common method used to achieve this is to generate stereo image pairs, one for the left eye and one for the right eye, that are displaced in the same way as when a real object is floating in front of the viewer. In stereoscopic HMD systems, these stereo pairs are displayed on two separate displays, using either cathode-ray tubes (CRT) or liquid crystal displays (LCD), mounted in front of each eye. Based on the position and
orientation of the viewer’s head, the left and right projections of the VE are calculated and rendered onto their respective displays.

In HTD systems the stereo image pairs are rendered onto one and the same display. Here the viewer wears a device that separates the left and right eye images into the correct eye. This separation must be performed accurately. If not, images intended for one eye may “bleed through” to the other which makes it difficult for the human brain to fuse the image pairs into one stereoscopic image. The most common methods to separate image pairs into the correct eye are through bichromatic image pairs, time sequential frames or image polarisation. In bichromatic stereo, the left and right eye views are each rendered in a different colour (often blue/red or green/red) while the viewer wears glasses with the same coloured filters in front of each eye. The filters absorb light of the same colour while passing the other. A major disadvantage of this technique is that all colour information in the images is lost. Time sequential systems render left and right eye views in rapid succession after each other in time while “shutter glasses” block the image for one eye and allow it to pass for the other. As long as the frequency at which this takes place is higher than 50 cycles per second, the viewer will see a flicker free stereoscopic image. Yet another technique is to use filters that change the polarization direction of the light emitted by the display differently for each image. The most common system use linear polarization filters placed in an orthogonal configuration (for example; up/down for the left eye and left/right for the right eye). In this case, the viewer wears polarized filters that only pass light that is polarized in the same direction as the filter. An advantage of this technique is that these (passive) polarized glasses are relatively cheap compared to (active) shutter glasses and that the images are flicker free. A disadvantage is that the viewer should always keep his head in line with the polarization direction of the emitted light. If the viewer tilts his head, images intended for one eye will bleed through to the other, destroying the stereo effect. Circular polarization filters do not have this problem, but unfortunately these filters are more expensive than linear polarization filters while projection screens that do not distort circular polarization are even costlier. Note that the techniques described here all block half of the light emitted by the displays. As a result, the viewer always perceives an image that is half as bright compared to the original.

The decision on which type of display is “best” for a specific application depends mainly on the indirect consequences of the categories. HMD devices isolate the viewer from the real world, in particular the types that are completely closed. The viewer will not be able to see his office, his colleagues, or even his own hands. In HTD systems, the viewer is still able to see objects in the physical world which make them more suitable in collaborative applications. If more than one person should be able to take part in the virtual experience, a HMD based setup requires that each viewer wears a display, possibly with its own graphics system. In HTD systems more than one person can share the same display space and make use of relatively cheap glasses. HMD systems require less space than HTD systems but HMD system can also be quite heavy to wear. Also, HMD systems suffer more from the effect of delays in the system than HTD systems. As there is always some delay between the moment the viewer moves
his head and the moment the newly calculated image appears, HMD systems often suffer from "lag". This lag is caused by delays in obtaining information on the head's new position and orientation and the recalculation of the new images. Depending on the length of these delays, the viewer can feel disorientated or even nauseated, a phenomenon that resembles see-sickness but in the context of VR research it is sometimes referred to as "cyber sickness".

A recent development in VR display technology are autostereoscopic displays. These devices are based on LCD displays, overlayed with an optical prism sheet that is aligned in such a way so that a given column of pixels is only seen by one eye, and not the other. The obvious benefit of these displays is that the viewer does not need to wear any eye-wear. However, as a result of the technique, the effective resolution is greatly diminished. Also, the displays often require the viewer to remain within the frustum in which the stereoscopic effect can be seen.

VR interaction technology

A vital ingredient in the development of interactive VR are the input devices that sense position, posture and gestures of the user. Tracking sensors form an integral part of many VR systems developed over the years and are most often used to track the position and orientation of the user's head and hand at the least. An important step in the development of unencumbering devices that could track the position and orientation of a physical object was taken in 1979 by Raab et al. with the development of an electromagnetic sensor known as the "Polhemus" [192]. These sensors are capable of measuring position and orientation at six degrees of freedom (DOF), three translational and three rotational, with reasonable accuracy.

Electromagnetic tracking systems use a transmitter that contains three orthogonal coils that are energized sequentially first for the x axis, then y, then z. The currents through the coils generate a magnetic field that generates a current in the receiver which also contains three orthogonal coils. When each transmitter coil is energized, the current through all three receiver coils is measured. This results in 3 receiver values for each of the 3 transmitter coils. The result is a system of linear equations that have enough unknowns to resolve the position as well as the orientation of the receiver relative to the transmitter.

Today, there are still mainly two companies that produce magnetic tracker products: Polhemus and Ascension, a company founded by Ernie Blood who was a former employee of Polhemus (both companies are located in Vermont) [253,257]. The main difference between the two types is their sensitivity to the vicinity of metal objects. The trackers developed by Ascension use a direct current (DC) to energize the transmitter coils rather than an alternating current (AC) as used in the Polhemus trackers. A DC current will, after an initial spike, generate no current in any surrounding metal instead of a constantly changing current with AC systems. Also, the Ascension devices turn the transmitter off between measurement cycles to take an additional set of readings to compensate for ambient electromagnetic fields. This combination makes the Ascension devices more accurate in environments with metal
objects. Tracking devices developed by other companies like InterSense [255] use different techniques, such as ultrasound, inertia, images obtained from cameras or hybrid techniques that use combinations of these. Critical specifications of tracking sensors are [128, 129, 160, 192] (see also Table 1.1):

- Resolution: the smallest displacement that can be measured by a sensor. In tracking systems where the position and orientation of a sensor is determined relative to a source, the resolution frequently depends on the distance of the sensor to the source.

- Accuracy: the deviation from the reported position and orientation of a sensor to its actual position and orientation. In tracking systems where the position and orientation of a sensor are determined relative to a source, the accuracy frequently depends on the distance of the sensor to the source.

- Update frequency: the number of updates the sensor is able to provide per second. In some cases the update frequency decreases as more sensors are used simultaneously in the same setup.

- Latency: the time lapse between a displacement of the sensor and the moment this is available at its output. The latency often depends on the type of interface between the tracking and the computing system (i.e. serial connection, ethernet, etc.).

- Jitter: the amount of reported displacement changes when the sensor is held still.

- Range: the volume in which the sensor can be used accurately. Note that some systems use technology that requires a line-of-sight between the source and the sensor.

- Drift: the accumulating error, over time, of the reported position and orientation of a sensor to its actual position and orientation. To compensate, sensors must either be recalibrated after a longer period of use or, otherwise, be augmented by other types of sensors.

1.5 The Cave Automated Virtual Environment

The Cave Automated Virtual Environment (CAVE) developed by DeFanti et al. at the Electronic Visualization Laboratory (EVL, University of Illinois at Chicago) is a fully-immersive, projection based VR device that was first presented at the ACM SIGGRAPH conference in 1992 [52, 53]. The CAVE installed at SARA in 1997 by Silicon Graphics and Pyramid Systems consists of the following components [205].
### Table 1.1: Typical specifications of different types of tracking systems [1, 128, 129, 160, 192, 253–257].

<table>
<thead>
<tr>
<th>DOF</th>
<th>Magnetic</th>
<th>Ultrasound</th>
<th>Optical</th>
<th>Inertial</th>
<th>Mechanical</th>
<th>GPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resolution: position</td>
<td>6.5 mm at 30.5 cm</td>
<td>0.5 cm</td>
<td>0.2 – 0.7 mm</td>
<td>N.A.</td>
<td>0.5 mm</td>
<td>5 m</td>
</tr>
<tr>
<td>Resolution: orientation</td>
<td>0.1 at 30.5 cm</td>
<td>5</td>
<td>0.01</td>
<td>0.02</td>
<td>0.1</td>
<td>N.A.</td>
</tr>
<tr>
<td>Accuracy: position</td>
<td>1.8 mm</td>
<td>5 cm</td>
<td>0.4 mm</td>
<td>N.A.</td>
<td>1 mm</td>
<td>2 m</td>
</tr>
<tr>
<td>Accuracy: orientation</td>
<td>0.5</td>
<td>5</td>
<td>0.02</td>
<td>3</td>
<td>0.5</td>
<td>N.A.</td>
</tr>
<tr>
<td>Update frequency</td>
<td>120 Hz</td>
<td>20-50 Hz</td>
<td>60 - 600 Hz</td>
<td>180-500 Hz</td>
<td>70 Hz</td>
<td>1 Hz</td>
</tr>
<tr>
<td>Latency</td>
<td>4-20 ms</td>
<td>60 ms</td>
<td>1 - 60 ms</td>
<td>2 ms</td>
<td>1 ms</td>
<td>2 s</td>
</tr>
<tr>
<td>Jitter</td>
<td>low</td>
<td>high</td>
<td>low-medium</td>
<td>low</td>
<td>low-medium</td>
<td>medium-high</td>
</tr>
<tr>
<td>Range</td>
<td>3 m</td>
<td>10 m</td>
<td>0.5 - 3 m</td>
<td>360° all axes</td>
<td>1-2 m</td>
<td>outdoors</td>
</tr>
<tr>
<td>Drift</td>
<td>medium</td>
<td>medium</td>
<td>low</td>
<td>high</td>
<td>low-medium</td>
<td>low</td>
</tr>
<tr>
<td>Cost</td>
<td>$6000</td>
<td>$500</td>
<td>$150 - $50k</td>
<td>$1000</td>
<td>$1500 - $95k</td>
<td>$500</td>
</tr>
</tbody>
</table>

#### 1.5.1 Computing and graphics hardware

The stereoscopic images in the CAVE are generated by an SGI Onyx2 RealityMonster [217]. The machine at SARA was one of the first delivered by SGI and, at the time of installation in 1997, the InfiniteReality hardware was among the most powerful general purpose graphics systems available. SARA’s CAVE configuration consists of the following computing and graphics components:

- 8 MIPS R10000 processors, each running at 195 MHz clock speed, each with 4 MB second level cache,
- 8 × 128 MB memory configured into 1 GB main memory using a cache-coherent Non Uniform Memory Access (ccNUMA) architecture,
- 4 InfiniteReality2 graphics pipes, each consisting of 4 Geometry Engines, 2 Raster Managers with 16 MB texture memory and a Display Generator,
- over 100 GB home file system on a dedicated O200 fileserver,
- HIPPI High speed network interface (800 Mbit/s),
- 8 channel digital audio system (ADAT). The CAVE uses four audio channels which allows front/back and left/right positioning of sounds.

The hardware architecture of the InfiniteReality2 graphics interfaces is configured into a pipeline. Each pipe can be subdivided into multiple channels (up to a maximum) where each channel is handled sequentially within one pipe. The performance of a channel is therefore roughly the performance of one pipe divided by the number of channels. The “Geometry Engines” provide the main interface to the main CPUs. They perform object transformation (scaling, translation, rotation), subdivide polygons into triangles and perform the projection of 3D world coordinates to 2D screen.
coordinates. The “Raster Managers” convert the projected triangles into pixel representation using a scan conversion technique. Finally, the “Display Generators” convert the digital pixel streams into an analog video signal that can be displayed by a monitor or projector.

1.5.2 Projection system

In order to fit the SARA CAVE on the available floorspace, foil mirrors are used to reflect the images onto the CAVE’s projection screens (see Figure 1.7). The projectors used in the SARA CAVE are Electrohome Marquee 8500 which are capable of handling resolutions up to a maximum of 1280 by 1024 pixels at 120 Hz. The CAVE uses an active system to produce stereo images. The graphics pipelines alternately generate the images for the left and right eye while the Stereographics CrystalEyes [49] liquid crystal shutter glasses that the viewers wear open and close synchronized with the images on the screen. This synchronization signal comes from a number of infrared emitters placed around the CAVE.

![Figure 1.7: Projectors, mirrors and screens setup of the CAVE used at SARA.](image)

Projection based VR systems have several advantages over “classical” HMD systems. First; projection based systems allow more than one person to share the same VE at minimal cost. In practice there are practical limits to the maximum number of people, which are often caused by the physical dimensions and placement of the projection screens. For example, although the floor area in the CAVE is 9 m², more than 5 people in the CAVE at the same time will quickly get in each other’s way. Of course,
1.5 The Cave Automated Virtual Environment

when multiple HMDs are properly connected together they could also be used to allow multiple persons to share the same environment, but the cost in additional hardware would be far greater than that of the additional shutter glasses in a projection system. Moreover, projection based systems do not obscure a human’s peripheral vision as HMD systems do. The user will therefore be able to see physical objects as well, most importantly his own limbs and fellow researchers, but also objects that can be useful while exploring the VE. This feature, however, also points us at a hindrance of projection based systems; virtual objects will be obscured from view by real objects if these are in the line of sight from the viewer to the projection of the virtual object on screen, even if the real object was really behind the virtual object. This can make interaction with VEs cumbersome in some situations. Also; the images in the CAVE are drawn in correct perspective only from the tracked user’s viewpoint. People next to this user see the same perspective, as if they were standing at the exact same location (see Figure 1.8). Since they are not, a user pointing his finger at a virtual object will seem to point at something completely different from a guest’s viewpoint [200]. To resolve both of these issues, a virtual pointer is often used, represented in the same virtual space. It therefore suffers from the same change in perspective as all other virtual objects. The stereoscopic representation of the virtual pointer, together with other depth cues such as partial occlusion caused by the intersection of the virtual pointer with the objects under investigation provide a sufficient solution to these problems.

\[\text{object location as seen by guest viewer} \quad \text{object location as seen by tracked viewer}\]

\[\text{left eye image} \quad \text{right eye image} \quad \text{projection screen} \quad \text{tracked viewer} \quad \text{guest viewer}\]

\[\text{Figure 1.8: Difference in perceived location of a virtual object as seen by the tracked viewer and a guest viewer.}\]
1.5.3 Interaction devices

To be able to generate the correct perspective view, the computer needs the position and orientation of the user's eyes. To measure this the CAVE uses an Ascension Technologies Flock of Birds (Extended Range) electromagnetic tracking system [45], with one of the sensors attached to the user's shutter glasses. A second sensor is attached to a "wand", a hand-held device with 3 buttons and a small joystick that is used to navigate through and interact with the virtual environment. Additional tracking sensors are available for application specific purposes. Other input devices that are in use (or have been used) in the SARA CAVE include a 15 sensor Ascension MotionStar Wireless full-body motion capture setup [46], a pair of CyberGloves [48], PC-joystick interfaces (joysticks, gamepads, etc.) that can be used via an Unwinder box [184] and a wireless microphone that can be used for audio-conferencing or in conjunction with SARA's speech recognition software (CAVE-Talk) which is based on IBM's ViaVoice package [47].

1.5.4 Writing CAVE applications

The software environment of choice for the development of CAVE applications is the "CAVE library" (CAVELib) [249]. CAVELib provides an application programming interface (API) that abstracts from the VR hardware and allows VR applications to run on systems ranging from HMDs to multi-walled projection based displays with little to no changes to the program. More information on the structure of CAVELib applications and an analysis of some of its design problems are provided in Appendix A. Other APIs that support CAVE-like environments include the WorldToolkit [212], Bamboo [251] and Avocado [237]. Open source and research initiatives (some of which include support for CAVE environments) include VRJuggler [241], FreeVR [216], DIVE [40], MAVERIK [101], PVR [244], and VIRPI [86].

1.6 Overview of this work

This thesis describes the technological, application and scientific issues involved in the design of environments for the purpose of interactive exploration in virtual environments.

Chapter 2 begins with an analysis of the situations in which virtual environments can provide significant benefits over traditional exploratory data analysis and describes the issues involved in the design and implementation of interactive exploration environments for use in virtual environments. These issues are illustrated by the description of a number of test cases.

Chapter 3 describes the design and construction of a low-cost virtual reality system, the Universiteit van Amsterdam Distributed Real-time Interactive Virtual Environment (UvA-DRIVE), built out of inexpensive, off-the-shelf hardware, and shows how this system can be used for interactive exploration at a fraction of the cost of similar commercially available systems.
Chapter 4 describes interaction methods for use in virtual environments to support scientific exploration. Interaction is at the center of any type of exploration. We describe several techniques that facilitate the construction of information-rich and highly interactive virtual environments.

Chapter 5 enhances the static design concepts towards the design of Interactive Dynamic Exploration Environments (IDEE). We will see that the design of these systems benefits from a distributed architecture where the various components execute on different systems and describe how these systems can be used for human-in-the-loop experimentation.

Chapter 6 describes an IDEE which was built as a test case to validate the ideas described in the previous chapters. This environment combines distributed simulation, interactive dynamic exploration and virtual reality technology into an interactive simulated vascular reconstruction operating theatre that allows pre-operative studies of abdominal vascular reconstruction procedures.

In Chapter 7 we recapitulate all issues addressed in this thesis and reflect on the design decisions that have been taken in the progress of this work. Directions for future research are described as well as recent scientific and technological developments that are relevant to this work.
Interactive exploration environments